ISSN 2676-9425

CENTRAL-EUROPEAN JOURNAL OF NEW TECHNOLOGIES IN RESEARCH, EDUCATION AND PRACTICE

Volume 1

Number 1

2019

Eötvös Loránd University
Faculty of Informatics
Budapest, Hungary

Contents

Programming Theorems Have the Same Origin

SZLÁVI Péter, TÖRLEY Gábor, ZSAKÓ László

Abstract: One of the classical ways of learning programming is to divide programming tasks into large groups, so-called programming theorems, and then to trace the specific tasks back to the programming theorems. Each teaching method introduces a different amount of programming theorems into the learning process, occasionally even combining them. In this article we will show that the basic and complex programming theorems have the same origin; consequently, it would be enough to present one theorem and trace everything back to it. At the end of the article, then, we will explain the practical use of still introducing more theorems.

Keywords: programming theorem, programming methodology, methodical programming, algorithm, specification

1. Introduction

In order to be able to solve a programming task, we need to identify its essence: programming tasks can be categorized into groups according to their type, which is useful because for each group we can create an algorithm rule that solves all of the tasks in that specific group. These task types are called programming theorems because their solutions are justifiably the correct solutions. Their number can vary from teaching method to teaching method. [1,2,3,4]

If we are familiar with the programming theorems, all we have to do to solve most tasks is to recognize the suitable programming theorem, use the specific data of the general task type, and in the general algorithm substitute them with the task-specific data. Applying this method is supposed to lead us to the correct solution. [5,6]

In these tasks we usually have to assign a certain result to one (or more) data collection(s), which, for simplicity's sake, we will handle as some sort of sequences. The essence of a sequence is the processing order of the given elements. Most of the times, it is sufficient to deal with sequences whose elements can be processed – one by one – one after another. In the input sequence, this requires an operation that is capable of giving the elements of the sequence one by one, while in the output sequence elements can be followed by a new element. In simple cases sequences can be illustrated as arrays.

The basic programming theorems are those that assign one value to one sequence: [7]:

- sequential computing
- counting
- maximum selection
- decision
- selection
- search

2. The first programming theorem

The first programming theorem comes from a simple task, that of calculating the sum of numbers. In the following, we will provide the specification and the algorithm of the task, which are the two pillars of the programming theorem. We are applying the marking system from [5,6].

Input: $N\in\mathbb{N}$, $X\in H^N$, $\Sigma:H^*\rightarrow H$, $+:H\times H\rightarrow H$ (where $H=\mathbb{N}$ or $H=\mathbb{Z}$ or $H=\mathbb{R}$)
$\Sigma(X_1,...,X_N)=\Sigma(X_1,...,X_{N-1})+X_N$, $F()=0$

Output: $S\in H$

Precondition: —

Postcondition: $S=\Sigma(X_1,...,X_N)$

```
Sum(N,X,S):
  S:=0
  For i:=1 to N do
    S:=S+X[i]
  End For
End.
```

Generalizing this task, we will get the so-**called sequential computin**g programming theorem. We generalize the operation +:

1) the generalized, binary (f) operation should have a (left-side) zero element (F0);

2) when we apply the operations one after another, the result should not depend on the execution order; that is, the operations should be associative.

The operation, based on binary operation f and zero element F0, and interpreted at H*, is indicated with F.

Input: $N\in\mathbb{N}$, $X\in H^N$, $F:H^*\rightarrow H$, $f:H\times H\rightarrow H$, $F_0\in H$
$F(X_1,...,X_N)=f(F(X_1,...,X_{N-1}),X_N)$, $F()=F_0$

Output: $S\in H$

Precondition: —

Postcondition: $S=F(X_1,...,X_N)$

Note: many times F is Σ, average, deviation, scalar multiplication, Π, ∪, ∩, ∀, ∃, writing one after another, Max, Min.

```
Computing(N,X,S):
  S:=F0
  For i:=1 to N do
    S:=f(S,X[i])
  End For
End.
```

Note: we are indicating changes from the previous algorithms with **bold**, both here and from now on.

3. Counting

Sequential computing can be formulated so that its result is the sum of all elements with T feature; thus, leading us to the **counting** programming theorem.

Function F will be a **conditional sum**. This means that we define function f as a **conditional function**, whose value is the first parameter+1 if the second parameter is of T feature, otherwise it is the value of the first parameter.

Input: $N\in\mathbb{N}$, $X\in H^N$, $F:H^*\to\mathbb{N}$, $f:\mathbb{N}\times H\to\mathbb{N}$, $F_0\in\mathbb{N}$

$$F(X_1,\ldots,X_N)=\begin{cases}F(X_1,\ldots,X_{N-1})+1 & \text{if } T(X_N)\\ F(X_1,\ldots,X_{N-1}) & \text{otherwise}\end{cases},$$

$F()=F_0$

$$f(a,b):=\begin{cases}a+1 & \text{if } T(b)\\ a & \text{otherwise}\end{cases},$$

$F_0=0$

Output: $Count\in\mathbb{N}$

Precondition: —

Postcondition: $Count=F(X_1,\ldots,X_N) \to Count=\sum_{i=1}^{N}{}_{T(X_i)}\,1$

```
Counting(N,X,S):
  Count:=0
  For i:=1 to N do
    Count:=f(Count,X[i])   →   If T(X[i]) then Count:=Count+1*
  End For
End.
```

Note: the change marked with * is the normalized algorithmic transformation of the conditional expression.

4. Maximum selection

If we replace function **f** with function **max**, we will get to one of the versions of the **maximum selection** theorem.

Input: $N\in\mathbb{N}$, $X\in H^N$, $F:H^*\to H$, $max:H\times H\to H$, $F_0\in H$, (H ordered set)
$F(X_1,\ldots,X_N)=max(F(X_1,\ldots,X_{N-1}),X_N)$,
$F()=F_0$

$$max(a,b)=\begin{cases}a & \text{if } a\geq b\\ b & \text{otherwise}\end{cases},$$

$F_0=-\infty$

Output: $MaxVal\in H$

Precondition: $N>0$

Postcondition: $MaxVal=F(X_1,\ldots,X_N)$

```
Maximum(N,X,MaxVal):
  MaxVal:=-∞
  For i:=1 to N do
    MaxVal:=max(MaxVal,X[i])  →  If X[i]>MaxVal then MaxVal:=X[i]
  End For
End.
```

With a very simple modification, we can even define the version that provides the maximum index as well. The output, the post-condition and the algorithm change as follows:

Output: MaxVal∈H, MaxInd∈ℕ

Postcondition: MaxVal=F(X_1,...,X_N) and 1≤MaxInd≤N and X_{MaxInd}=MaxVal

```
Maximum(N,X,MaxVal,MaxInd):
  MaxVal:=-∞
  For i:=1 to N do
    If X[i]>MaxVal then MaxVal:=X[i]; MaxInd:=i
  End For
End.
```

If we make use of the precondition, we can leave out the check of the first element from the iteration; in fact, in the classical version we do not need the maximum value either. Hence, the final version is like this:

```
Maximum(N,X,MaxInd):
  MaxInd:=1
  For i:=2 to N do
    If X[i]>X[MaxInd] then MaxInd:=i
  End For
End.
```

5. Decision

In sequential computing, function **F** can be operator ∃ too, which leads us to the **decision** theorem. The operator 'or' is associative, its zero element is 'false'; thus, we can include it in the basic version.

Input: N∈ℕ, X∈H^N, ∃:H^*→L, T:H→L, or:L×L→L, F_0∈L
F(X_1,...,X_N)=∃i(1≤i≤N): T(X_i)
∃i(1≤i≤N): T(X_i) ≡ ∃i(1≤i≤N-1): T(X_i) or T(X_N),
f(a,b)=a or b, F()=F_0=false

Output: Exists∈L

Precondition: —

Postcondition: Exists=∃i(1≤i≤N) T(X_i)

```
Decision(N,X,Exists):
  Exists:=false
  For i:=1 to N do
    Exists:=Exists or T(X[i])
  End For
End.
```

Note: in the iteration the variable **Exists** can change in the following way:

- `false, ..., false, true, ..., true`
- `false, ..., false`

that is, either it remains to be false all through, or it becomes true and stays like that. Consequently, once it becomes true, the iteration can stop. Before it does, it is constantly false, so we do not need to change value. The decision programming theorem derives from this: by the end of the iteration we will be able to distinguish which of the two cases we are dealing with.

Input: $N \in \mathbb{N}$, $X \in H^N$, $T:H \rightarrow L$

Output: $Exists \in L$

Precondition: —

Postcondition: $Exists = \exists i (1 \leq i \leq N)\ T(X_i)$

```
Decision(N,X,Exists):
  i:=1
  While i≤N and not T(X[i])
    i:=i+1
  End While
  Exists:=(i≤N)
End.
```

6. Selection, Search

At the end of the decision iteration, the value of variable i is the ordinal number of the item of feature T, provided that we know such an item exists (that is, the iteration will stop once an item is found) → **Selection** theorem

Input: $N \in \mathbb{N}$, $X \in H^N$, $T:H \rightarrow L$

Output: $S \in \mathbb{N}$

Precondition: $\exists i (1 \leq i \leq N)\ T(X_i)$

Postcondition: $1 \leq S \leq N$ and $T(X_S)$

```
Selection(N,X,S):
  i:=1
  While ~~i≤N and~~ not T(X[i])
    i:=i+1
  End While
  S:=i
End.
```

The ordinal number of an element of T quality is N+1 if it has not gone beyond the end of the sequence. If it has → **Search** theorem

Input: $N \in \mathbb{N}$, $X \in H^N$, $T:H \rightarrow L$

Output: $Exists \in L$, $S \in \mathbb{N}$

Precondition: —

Postcondition: $Exists = \exists i (1 \leq i \leq N)\ T(X_i)$ and $Exists \rightarrow 1 \leq S \leq N$ and $T(X_S)$

```
Search(N,X,Exists,S):
  i:=1
  While i≤N and not T(X[i])
    i:=i+1
  End While
  Exists:=(i≤N)
  If Exists then S:=i
End.
```

7. Complex programming theorems

Complex programming theorems belong to problems that assign sequence(s) to sequence(s) [8]:

- copying,
- multiple item selection,
- partitioning,
- intersection,
- union.

8. Copying – function calculation

We can also define the theorem of sequence computing by applying function (**g**) to each element of the input array and placing the element (**f**) at the end of the output array (**Y**). This way, the null element ($\mathbf{F_0}$) of Y will be the empty array, and function **F** will add the elements of $\mathbf{g(X_i)}$ to the elements of **Y.** Practically, function **F** will be an addition operation interpreted for the array (which is what have signaled below with operation „push back").

Input: $N \in \mathbb{N}$, $X \in H^N$, $g:H \rightarrow G$, $F: G^N \rightarrow G$, $f: G^* \times G \rightarrow G$, f – push back

Output: $Y \in G^N$

Precondition: –

Postcondition: $\forall i(1 \leq i \leq N): Y=F(g(X_1),\ldots, g(X_N))$

```
Copying(N,X,Y):
  Y:=empty
  For i:=1 to N do
    Y:=push_back(Y,g(X[i]))
  End For
End.
```

If we assigned **string** type variables to the input and output arrays, the above algorithm would look like this:

```
Copying(N,X,Y):
  Y:=""
  For i:=1 to N do
    Y:=Y+g(X[i])
  End For
End.
```

This refers even more to sequence computation.

Since each element of **Y** will be $\mathbf{g(X_i)}$, the theorem can be expressed like this as well (we are indicating function **g** from the previous algorithm with **f** now):

Input: $N \in \mathbb{N}$, $X \in H^N$, $f:H \rightarrow G$

Output: $Y \in G^N$

Precondition: –

Postcondition: $\forall i(1 \leq i \leq N): Y_i=f(X_i)$

```
Copying(N,X,Y):
  For i:=1 to N do
    Y[i]:=f(X[i])
  End For
End.
```

Finally, we will arrive to the theorem of conditional copying with a simple modification, namely, that we apply function calculation only to elements with property T.

Input: $N \in \mathbb{N}$, $X \in H^N$, $f:H \to G$, $T:H \to \mathbb{L}$

Output: $Y \in G^N$

Precondition: –

Postcondition: $\forall i(1 \leq i \leq N)\ T(X_i) \to Y_i = f(X_i)$ and not $T(X_i) \to Y_i = X_i$

```
Copying(N,X,Y):
  For i:=1 to N do
    If T(X[i]) then Y[i]:=f(X[i]) else Y[i]:=X[i]
  End For
End.
```

9. Multiple item selection

In sequence computing we may substitute function **F** with conditional addition interpreted for the array, which will lead us to the theorem of multiple item selection. $\mathbf{X_i}$ will be added to array **Y** if it has property **T**, otherwise it is not added (that is, we add nothing).

Input: $N \in \mathbb{N}$, $X \in H^N$, $T:H \to \mathbb{L}$

Output: $Y \in H^*$

Precondition: –

Postcondition: $\forall i(1 \leq i \leq N)$: $T(X_i) \to X_i \in Y$ and not $T(X_i) \to X_i \notin Y$

```
Multiple_item_selection(N,X,Y):
  Y:=empty
  For i:=1 to N do
    If T(X[i]) then Y:=push_back(Y,X[i]) {else nothing to do}
  End For
End.
```

Starting from counting:

```
Counting(N,X,Count):
  Count:=0
  For i:=1 to N do
    If T(X[i]) then Count:=Count+1
  End For
End.
```

Connecting the two algorithms:

```
Multiple_item_selection(N,X,Count,Y):
  Count:=0
  For i:=1 to N do
    If T(X[i]) then Count:=Count +1; Y[Count]:=X[i]
  End For
End.
```

Or a different approach:

Input: $N \in \mathbb{N}$, $X \in H^N$, $T:H \to \mathbb{L}$,
$f:H^* \times H \to H$,
$f(x,e):=\begin{cases} push\ back(x,e), & T(e) \\ x, & otherwise \end{cases}$

Output: $Y \in H^*$

Precondition: –

Postcondition: $\forall i(1 \le i \le N)$: $Y_i = f(X, X_i)$

```
Multiple_item_selection(N,X,Y):
  Y:=empty
  For i:=1 to N do
    Y:=f(Y,X[i])
  End For
End.
```

We arrive to the final version after inserting the core of function f:

```
Multiple_item_selection(N,X,Y):
  Y:=empty
  For i:=1 to N do
    If T(X[i]) then Y:=push_back(Y,X[i]) else {nothing to do}
  End For
End.
```

10. Partitioning

The previously mentioned operation of conditional addition interpreted for the array will be applied with two conditions now. If $\mathbf{X_i}$ has property **T,** then we add it to array **Y**, otherwise to array **Z.**

Input: $N \in \mathbb{N}$, $X \in H^N$, $T:H \to \mathbb{L}$

Output: $Count \in \mathbb{N}$, $Y,Z \in \mathbb{N}^*$

Precondition: –

Postcondition:

$$Count = \sum_{\substack{i=1 \\ T(X_i)}}^{N} 1$$

and $\forall i(1 \le i \le Count)$: $T(X_i)$ and $\forall i(1 \le i \le N\text{-}Count)$: not $T(X_i)$ and $Y \subseteq (1,2,\ldots,N)$ and $Z \subseteq (1,2,\ldots,N)$

```
Partitioning(N,X,Count,Y,Z):
  Count:=0
  For i:=1 to N do
    If T(X[i]) then Count:=Count+1; Y[Count]:=i
  End For
  CountZ:=0
  For i:=1 to N do
    If not T(X[i]) then CountZ:=CountZ+1; Z[CountZ]:=i
  End For
End.
```

In the above algorithm, independent loops with the same amount of steps or branches with the same conditions can be handled together. This is how we get to the well-known algorithm of sorting in two.

```
Partitioning(N,X,Count,Y,Z):
  Count:=0; CountZ:=0
  For i:=1 to N do
    If T(X[i]) then Count:=Count+1; Y[Count]:=i
               else CountZ:=CountZ+1; Z[CountZ]:=i
  End For
End.
```

Therefore, sequential computing led to copying, copying led to multiple item selection, and multiple item selection led to partitioning; in short, even partitioning originates from the theorem of computing.

11. Intersection (decision in multiple item selection)

The theorem of intersection is essentially the combination of two theorems, that of multiple item selection and that of decision. We have already proven that both derive from the theorem of sequential computing.

Input: $N,M\in\mathbb{N}$, $X\in H^N$, $Y\in H^M$

Output: $Count\in\mathbb{N}$, $Z\in H^{Count}$

Precondition: isSet(X,N) and isSet(Y,M)

Postcondition:

$$Count = \sum_{\substack{i=1 \\ X_i\in Y}}^{N} 1$$

and $\forall i(1\leq i\leq Count)$: $Z_i\in X$ and $Z_i\in Y$ and isSet(Z,Count)

Definition: $isSet:H^*\times\mathbb{N}\rightarrow\mathbb{L}$

$isSet(h,n)=\forall i,j(1\leq i\neq j\leq n): i\neq j\rightarrow h_i\neq h_j$

```
Intersection(N,X,M,Y,Count,Z):
  Count:=0
  For i:=1 to N do
    If (X[i] in Y) then Count:=Count+1; Z[Count]:=X[i]
  End For
End.
```

```
operator in(x,Y):
  j:=1
  While j≤M and x≠Y[j]
    j:=j+1
  End While
  in:=(j≤M)
End.
```

After inserting the function:

```
Intersection(N,X,M,Y,Count,Z):
  Count:=0
  For i:=1 to N do
    j:=1
    While j≤M and X[i]≠Y[j]
      j:=j+1
    End While
    If j≤M then Count:=Count+1; Z[Count]:=X[i]
  End For
End.
```

12. Union (copying + decision in multiple item selection)

Practically, the theorem of union is the blend of the copying, multiple item selection, and decision theorems. It has already been demonstrated that all three are based on the theorem of sequential computing.

Input: $N,M\in\mathbb{N}$, $X\in H^N$, $Y\in H^M$

Output: $Count\in\mathbb{N}$, $Z\in H^{Count}$

Precondition: isSet(X,N) and isSet(Y,M)

Postcondition:

$$Count = N + \sum_{\substack{j=1 \\ Y_j\in X}}^{M} 1$$

and $\forall i(1\leq i\leq Count)$: $Z_i\in X$ or $Z_i\in Y$ and isSet(Z,Count)

```
Union(N,X,M,Y,Count,Z):
  For i:=1 to N do
    Z[i]:=X[i]
  End For
  Count:=N
  For j=1 to M do
    If not (Y[j] in X) then Count:=Count +1; Z[Count]:=Y[j]
  End For
End.
```

After inserting the function and assigning value to the arrays:

```
Union(N,X,M,Y,Count,Z):
  Z:=X; Count:=N
  For j=1 to M do
    i:=1
    While i≤N and X[i]≠Y[j]
      i:=i+1
    End While
    If i>N then Count:=Count+1; Z[Count]:=Y[j]
  End For
End.
```

13. Conclusion

From the above analysis, we can see that the first 6 theorems (sequential computing, counting, maximum selection, decision, selection, and search) all originate from one programming theorem, namely sequential computing, which is simple addition.

Furthermore, it was proven that 5 additional complex theorems (copying, multiple item selection, partitioning, intersection, and union) derive from the same theorem, that of computing, as well.

It is worth dedicating some time to uncovering why even complex theorems can be traced back to one of the simplest programming theorems. We can derive copying, multiple item selection, and partitioning back to sequential computing because copying applies addition interpreted for the array (that is, we start out from an empty array and while processing the elements of X, we add them to Y, to which we have referred with the operation "push back" in the theorem of copying). Multiple item selection is different from this only in the fact that we apply conditional addition. (The relation is similar to summing and conditional summing as explained in our previous article.) If multiple item selection can be traced back to computing, so can partitioning, and since intersection and union are based on the above, even those theorems are proven to originate from sequential computing.

Among complex theorems, we have not dealt with sorting. It is so because the specific algorithms of sorting can be traced back to different programming theorems, for example minimum selection sort comes from minimum selection and copying, insertion sort comes from search and (while) copying, counting distribution sort comes from counting and copying, etc.

Despite all the above, we are not saying that it is useless to manage these theorems independently. Beginner programmers will recognize the programming theorems while dealing with the different task types, which, in the case of basic theorems, correspond to the above defined six, while with complex theorems, they are most likely the above five. [1,5] With more advanced programmers, it would be worth to base the theorems not on the task types but on the solution types. In that case, decision, selection and search are three sub-types of the same solution principle; therefore, they can be considered as one programming theorem. [2]

This article was supposed to demonstrate that theoretically it is sufficient to check the validity of the first programming theorem because all the rest can be deducted from it. In sum, if sequential computing is correct, so are the others.

Bibliography

1 Péter Szlávi, László Zsakó: *Módszeres programozás. (Methodical programming)* Műszaki Könyvkiadó, Budapest. 1986.

2 Tibor Gregorics: *Programozás – Tervezés. (Programming – Design)* ELTE-Eötvös Kiadó, 2013.

3 Ákos Fóthi: *Bevezetés a programozáshoz. (Introduction to programming)* Fóthi Ákos, 2012. (http://people.inf.elte.hu/fa/pdf/konyv.pdf − Last Retrieved 05/05/2017)

4 Tibor Gregorics: *Programming theorems on enumerator*. Teaching Mathematics and Computer Science 8/1, 2010. DOI: 10.5485/TMCS.2010.0243 (http://tmcs.math.unideb.hu/load_doc.php?p=186&t=abs − Last retrieved 05/05/2017)

5 Péter Szlávi, László Zsakó at al.: *Programozási alapismeretek. (Programming basics)* online course material, (http://progalap.elte.hu/downloads/seged/etananyag/ - Last retrieved 05/05/2017), ELTE Informatikai Kar, 2012.

6 Péter Szlávi, László Zsakó at al.: *Módszeres programozás: programozási tételek*. (*Methodical programming: programming theorems*) Mikrológia 19. ELTE Informatikai Kar, 2008.

7 Péter Szlávi, Gábor Törley, László Zsakó: *Programming theorems have the same origin*. XXX. Didmattech 2017, Trnava University, Faculty of Education, 2017 (http://real.mtak.hu/55421/1/szp_tg_zsl_programming_theorems_have_the_same_origin.pdf − Last retrieved 05/05/2017)

8 Péter Szlávi, Gábor Törley, László Zsakó: *Az összetett programozási tételek is egy tőről fakadnak. (Complex theorems have the same origin too)* Infodidact2017, Webdidaktika Alapítvány, 2017 (http://people.inf.elte.hu/szlavi/infodidact17/manuscripts/zsltgszp.pdf − Last retrieved 02/18/2018)

Authors

SZLÁVI Péter

Eötvös Loránd University, Faculty of Informatics, Department of Media and Educational Informatics, Hungary, e-mail: szlavip@elte.hu

TÖRLEY Gábor

Eötvös Loránd University, Faculty of Informatics, Department of Media and Educational Informatics, Hungary,
e-mail: pezsgo@inf.elte.hu

ZSAKÓ László

Eötvös Loránd University, Faculty of Informatics, Department of Media and Educational Informatics, Hungary,
e-mail: zsako@caesar.elte.hu

About this document

Published in:

CENTRAL-EUROPEAN JOURNAL OF NEW TECHNOLOGIES IN RESEARCH, EDUCATION AND PRACTICE

Volume 1, Number 1. 2019.

ISSN: 2676-9425 (online)

DOI:

10.36427/CEJNTREP.1.1.380

License

Real-time Tools in Classroom[1]

BAKONYI Viktória, ILLÉS Zoltán, Dr. ILLÉS Zoltán

Abstract*:* Today we are surrounded by IT devices and a lot of different applications from which more and more are usually work in real-time. These systems are mainly embedded systems but there are general operating systems which offers such possibilities too. Moreover, we can talk about real-time applications on the web as well. Therefore, it became very important for each informatician especially for future informatics teachers to know about the bases of this technology. Everybody knows the fact that each new technology appears sooner or later in education and finds its place in it. These new possibilities may give additional tools into teachers' hands. The question is whether they can live with the opportunity or not.

Keywords*:* real-time systems, teachers' training, classroom response systems

1. Introduction

Our public life is already supported by applications. We may follow the tracks of airplanes, trains, buses, use chatbots as customers, even we may meet with robot waiters (2018 - https://bit.ly/2IhjUTw) or receptionists (2018 - https://bit.ly/2WawF70). For today autonomous cars became reality but real-time functionalities are built in normal cars too. There are more and more smart houses which are able to collect data about the environment or react to our requests in real-time. People are planning to build smart cities, which are able to handle common resources and meanwhile take care of the well-being of humans. It seems that we are living in the world of IoT (Internet of Things).

All of the previously mentioned possibilities describe computer-computer or human-computer interactions. Is there any newness in human-human interactions? Almost everybody has a smart phone with which people are able to speak with each other without any lagging. We use social networks to keep in contact or to get help in any problem we are facing to. What about education?

"*Students inhabit a 21st-century world for 18 hours a day, and, all too often, educators put them in a 19th-century classroom for six hours of that day, and the students feel a tremendous disconnect. We have a responsibility to teach them the skills to optimize these tools.*" (https://bit.ly/2FNB3m5)

Naturally, they must learn lifelong to keep on with their professions, but we have to offer a base on which they can build their future knowledge. Real-time world is around us, so we have to explain the ideas of real-time application features [1, 2, 3]. This technology appeared in the education as well therefore we can present RT educational tools and their usefulness. [4,5,6,7]

2. Real-time systems

2.1. The Notion

Before anything else we have to explain what, we mean under an RT system. Usually people think that a real-time system must be a very quick system, but it is a disillusion. The key is not the quickness of the operation but the existence of a deadline. It means that a system is real-time if the

[1] EFOP-3.6.3-VEKOP-16-2017-00001: Talent Management in Autonomous Vehicle Control Technologies – The Project is supported by the Hungarian Government and co-financed by the European Social Fund.

response must finish within the given deadline. If no delay is acceptable then it is called a ***hard-real time*** system (a delay may cause a catastrophe) otherwise it is a ***soft real-time*** system. It is quite simple to understand and define it but how can we develop such systems? The solution is not easy at all!

2.2. Architecture

Frequently real-time systems are running on embedded systems which are created for a special goal. A typical RT system is a controlled system which collects data through its sensors and controls the environment with its actuators warranting the deadline of the response. (Figure 1.)

Figure 1. Model of a controlled system

Usually it executes only one process, which collects data and may give some kind of responses within a deadline, but nowadays general operating systems offer RT features too.

Classification of RT systems:

- Single task – running on a device without OS (Naturally the programmer has to know the detailed information about the execution time of a single instruction, the accuracy of the system clock, the IO operations and the timer.)
- Single task – running on a device with OS (not too difficult to calculate the execution time of a program, similar to the previous)
- Multitask – running on a device/computer with OS

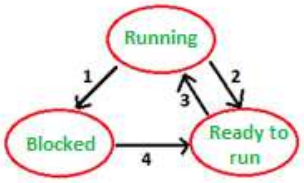


Figure 2 Processes in different states

- The real interesting question in the case of a multiuser, multitask OS is: how can we ensure the deadline running together several different tasks, which are a possibly mixed set of normal and real-time processes. Which of the processes should run next? (Figure 2.)
- The work is done by the schedulers. The common idea is to give some extra priority level to the real-time processes and so they can get more resources to run. Let us see how interrupt & priority handling may help.
- First the low priority interrupt arrives,
- Time for task switch - the interrupt handler has to be detected and loaded
- The lower priority handler just has started, but after a while a higher priority interrupt arrives with a close deadline. If the lower finishes the work, the deadline of the higher priority process is missed!
- Time for task switch - the higher interrupt handler has to be detected and loaded – the deadline is hold
- The lower priority handler may continue its work. Everything is fine!

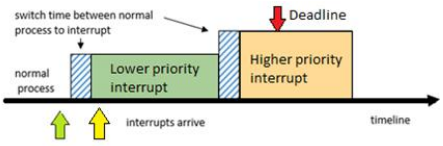


Figure 3. The deadline is missed

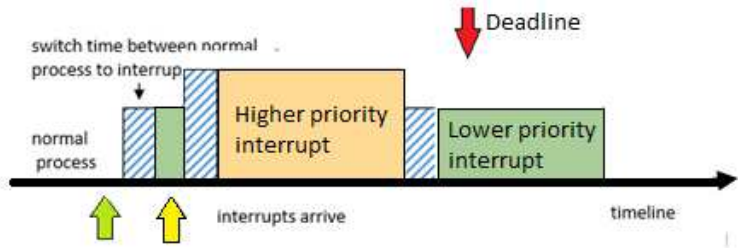


Figure 4. Nested (multilevel) interrupt handling

Naturally, the implementation is much more complicated, but the base idea(s) is shown in a subject named Operating Systems (5th semester in programmers training, and a few semesters later in teachers' training). There are different ideas of schedulers and we have to calculate the speed of timers, interrupts, the latency etc. The scheduling is more difficult if we have several processors in the computer.

In the case of Suse Linux Enterprise RT we can define so called cpusets and dedicate tasks to a given **cpuset** – if the cpuset consists of only one CPU, then we are back at the case of a unique processor… In the system there is always a root cpuset which is dedicated for the OS. We may create shield cpuset which is not used by the OS it may be used only for special dedicated tasks at it is shown on Figure 5.

```
os:/home/illes # cset set -r
cset:
         Name       CPUs-X    MEMs-X Tasks Subs Path
 ------------ ---------- - ------- - ----- ---- ----------
         root        0-3 y       0 y   525    0 /
os:/home/illes # cset shield --cpu=3
cset: --> activating shielding:
cset: moving 306 tasks from root into system cpuset...
[==================================================]%
cset: "system" cpuset of CPUSPEC(0-2) with 306 tasks running
cset: "user" cpuset of CPUSPEC(3) with 0 tasks running
os:/home/illes # cset shield --exec ls
cset: --> last message, executed args into cpuset "/user", new pid is: 23263
Desktop    Music      Templates  admin    oprendszer   pw_check.c
Documents  Pictures   Videos     bin      public_html  pw_check_print
Downloads  Public     a.out      kicsap   pw_check     tmp
os:/home/illes #
```

Figure 5. Usage of cset, shield in SuSe Linux

2.3. Real-time Web – Notion

If you understood the above described real-time notion you hardly believe that it may work in the case of web applications. You are right, but the real-time web has a different meaning! An RT web application does not need clients' requests to refresh the pages, while a classic web application does. Now there are several techniques the programmer may use to support RT web functionality like long polling, web sockets, etc. Using these technics, an RT web application can join to a service hub getting notifications from other participants without direct refreshing.

Such types of applications are used widely, e.g. the ARS (Audience Response System) is one of the most well-known. These types of systems are used to serve group meetings, virtual conference participations etc.

Sure, this type of applications can be used in education as well.

2.4. Benefits of Using a Classroom Response System

There are a lot of ready-made, platform independent, free of charge (mainly only part of it) CRS on the market. E.g. Kahoot (https://kahoot.com/),Socrative (https://www.socrative.com/),

Spiral (https://spiral.ac/), VoxVote (http://www.voxvote.com/) and Sli.do (https://www.sli.do/).

Our question should be why they are so popular in modern schools. The answer is quite complex as usual in real life. First today children are digital natives they *like* and always use their mobile devices as we emphasized previously. Remember the idea of Comenius learning should not be a task but a source for pleasure. We all know that *doing* something is much more effective than to read or listen to it – it was shown by E. Dale. Therefore, using a CRS in class each of the students is forced to deal with the topic. Practiced teachers know that nobody is able to pay attention too long, so some changings in activity may help to avoid *mind wandering*. *Formative evaluations* can be executed quickly, and we all know the benefits of them from the viewpoints of the teachers and the students as well. Meanwhile children may feel the *personal attention* of the teacher who may give immediate feedbacks to the results and it can mean a good motivation for them. Students, even shy ones are brave enough to *involve* them into the common work if the system gives anonymous possibilities. Furthermore, CRS helps in *self-assessment* as well – it makes possible to compare our own knowledge to the others'. With a modern CRS the teachers are able to collect data and make some data analyses later to enhance the effectivity of our teaching methods.

2.5. E-Lection

In our teaching practice we saw, the usage of such a CRS system is evident therefore we decided to integrate one of them to help students. As we started to use such a web-based RT Classroom Response System adding more interactivity to the lectures, we noticed a lot of problems.

Since we are not able to buy ready-made devices for each of the seats in the auditoriums, therefore we wanted to use the students' own devices. Moreover, we had a lot of preconditions (configurable authorization, post data analysing, etc) and they were missing from ready-made CRS system. So we had to implement an own CRS, called E-Lection which fits to our needs.

In the next some steps we describe our work.

1. Naturally, we had to make sure that the students have smart devices, that is why for preparation we made a survey in 2015/2016 academic year to explore the situation.

Survey about smart device usage

Circumstances: Google Form (to ensure the anonymity of students)

The survey available at http://bit.ly/1eGP5Hm (Hungarian)

Participants: 486 students from ELTE

Questions (interesting now): Do you have a mobile phone? Which type?

Table 1: Phone types – students' own

Own phone types	ELTE students
Smart phone	92,4 %
Android	68,4 %
Apple iPhone	0,3 %
Windows Phone	8,3 %
Other type	15,3 %
Normal	6,3 %
Nothing	1,3 %

The results showed that we are ready to use a BYOD system (Bring Your Own Device) over 92 percentage.

2. In the 2017/2018 academic year, in September we asked the students whether they think it is good to use a CRS system or not.

Survey about the idea of using a CRS

2.6. Precondition: Previously they did not use any CRS systems in ELTE. To fill the survey was not compulsory.

Circumstances: Google Form (to ensure the anonymity of students)
The survey available at https://bit.ly/2rcvR4P. (only in 0048ungarian)

Participants: 121 students who attended the 1st semester and 76 students who attended the 5th semester.

Precondition: Previously they did not use any CRS systems in ELTE. To fill the survey was not compulsory.

Questions: (all together 13+1)

a) Would you consider it benefitting if you could use your own mobile device (phone, laptop, etc.) during lectures? (grade: 1-5)
b) Would you consider it useful if bilateral communication happened between the lecturer and students? (grade: 1-5)

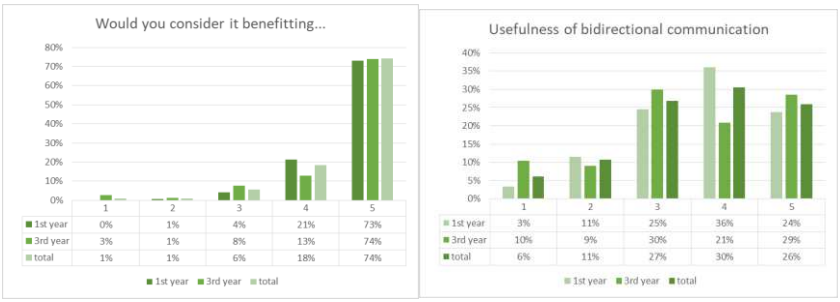


Figure 5. Results for question a and b.

The results showed that the students want to use their smart devices during lessons. More of them want to communicate and contact with teachers.

3. In 2017/2018 academic year, we implemented and tested our system - at first time using it in Operating system subject. This subject is a compulsory subject of programming informatics at the 5th semester. We compared the students' results with and without using E-lection in this year. (In the time of writing this paper, we do not have final results from 2018/2019 academic year.) We compared the results of this semester and the results of the previous year (2016/2017 autumn semester). [5]

Circumstances: each year we had at about 200 students.

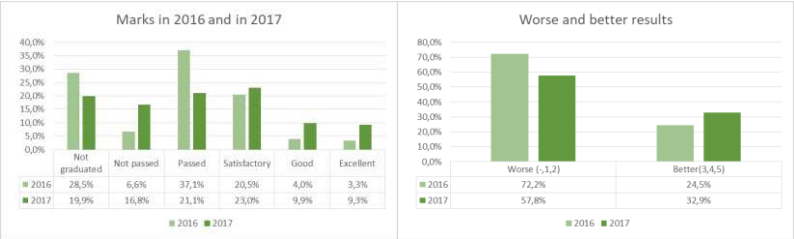


Figure 6. Compared results with and without E-Lection

As you can see on Figure 6, E-lection did not help too much for not graded or failed students, but the percentage of better results increased.

4. This academic year we retuned our system and tested again during Operating Systems subject. Usually the lectures started with one or two questions refreshing the basic ideas of the previous lesson and the students may answer them. In the middle of the semester we asked the students opinion about the usefulness of E-Lection.

Survey about the usefulness of E-Lection 2018/2019 academic year [7]

Circumstances: Google Form (to ensure the anonymity of students)

The survey available at *https://bit.ly/2CDsn10, https://bit.ly/2JycD5o* (English, Hungarian)

Participants: 84 students who filled the survey, Operating Systems (217 students all)

Questions: (There were more questions…)

a) In your opinion has E-Lection made classes more interesting? (grade: yes, no, sometimes)

yes	No	sometimes
73,8%	21,4%	4,8%

The result shows on Figure 1., that they enjoy our E-Lection system.

b) How helpful are the teachers' questions and their discussions in understanding the most important concepts and principals?

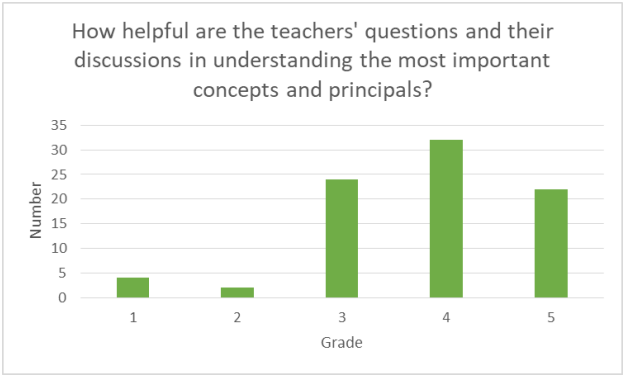


Figure 7. Usefulness of teachers' questions

5. To get a full panorama of CRS usage we decided to explore whether future teachers learn about CRS systems or not and what is the opinion of active teachers. We made a survey.

Survey about the usage of any kind of CRS, 2018/2019 academic year [4]

Circumstances: Google Form (to ensure the anonymity of students)

The survey is available at https://bit.ly/2KqLaBl (Hungarian)

Participants: 102 teachers/students who filled the survey,

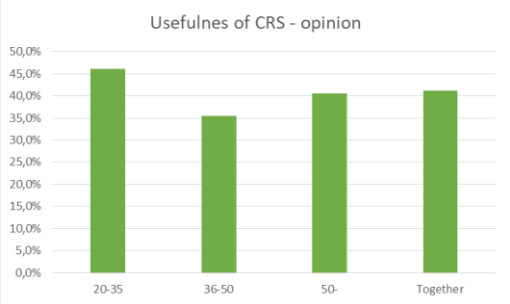
Questions: (There were more questions…)

a) The first thing we were interested in was that how many of them used CRS in school. The possible answers were never, rarely and often. (some of the persons skipped the question only 92 answers arrived.)

Age	Numb	Never	%	Rarely	%	Often	%
20-35	39	19	48,7	10	25,6	4	10,3
36-50	31	18	58,1	12	38,7	1	3,2
50-	32	20	62,5	6	18,8	4	12,5
Together	92	57	61,3	28	30,1	9	9,7

b) Do you think CRS is useful in school?

Age	Usefulnes
20-35	46,2%
36-50	35,5%
50-	40,6%
Together	41,2%



The result shows that teachers (over 35), who never learned about CRS in university are open to use new technologies and they keep on with IT technology! (It is true, that most of them were informatics teacher, therefore a general survey from human sciences may show another result.)

3. Conclusion

The spread of rich set real-time applications in daily life must change the focus points of education again. We must reorganize the content of subjects to present the newness of science and meanwhile we may use modern IT technology to make our teaching methods more inspiring. We implemented an own classroom system E-Lection and examined its reception and compared the achieved results. Our work is not finished yet, because we have to give new interesting features to the application according to the students' needs.

References

1 Illés, Z., Havancsák. K.: *Real-Time Computer Measurement Control under Dos-Windows Operating System.* In Annales Universitatis Scientiarum Budapestinensis de Rolando Eotvos Nominatae Sectio Computatorica XVII: pp. 193-199. (1998)

2 Tanembaum, A., Woodhull, A.: *Modern Operating Systems*, 2014. ISBN-13: 978-0133591620

3 Heizlerné Bakonyi, V, Illés, Z., Menyhárt, L. *Viewpoints for the development of teaching contents in the field of informatics* Acta Didactica Napocensia 6:(2) Paper 5. 10 p. (2013)

4 H. Bakonyi, V., Illés, Z.: *Real time classroom systems in teachers training* Lecture Notes in Computer Science 11169 : & pp. 206-215. , 10 p. (2018) DOI: 10.1007/978-3-030-02750-6_16

5 Bakonyi, V., Illés,Z. : *Experiences of Using Real-Time Classroom Response Systems* In: František, Jakab (szerk.) 16th IEEE International Conference on Emerging eLearning Technologies

and Applications : ICETA 2018, Košice, Szlovákia : Technical University of Kosice, (2016) pp. 51-56. , 6 p. DOI: 10.1109/ICETA.2018.8572042

6 H Bakonyi, V.; Illés, Z.: *Real Time Systems In Embedded and Enterprise World (2018), DidMatTech Conference, 2018, Radom*

7 H., Bakonyi, V.; Illés, Z.: *Valós idejű oktatási rendszerek,* In: Szlávi, Péter; Zsakó, László (szerk.) InfoDidact 2018, Budapest, Magyarország: Webdidaktika Alapítvány, (2019) pp. 51-58. , 8 p.

Authors

BAKONYI Viktória

Eötvös Loránd University, Faculty of Informatics, Department of Media and Educational Informatics, Budapest, Hungary, e-mail: hbv@inf.elte.hu

ILLÉS Zoltán, Ph.D.

Eötvös Loránd University, Faculty of Informatics, Department of Media and Educational Informatics, Budapest, Hungary e-mail: zoltan.illes@elte.hu

ILLÉS Zoltán

Eötvös Loránd University, Faculty of Informatics, Department of Media and Educational Informatics, Budapest, Hungary e-mail: ilzo@inf.elte.hu

About this document

Published in:

CENTRAL-EUROPEAN JOURNAL OF NEW TECHNOLOGIES IN RESEARCH, EDUCATION AND PRACTICE

Volume 1, Number 1. 2019.

ISSN: 2676-9425 (online)

DOI:

10.36427/CEJNTREP.1.1.381

License

Interdisciplinary Technical Exercises for Informatics Teacher Students

MAKAN Gergely, ANTAL Dóra, MINGESZ Róbert, GINGL Zoltán, KOPASZ Katalin, MELLÁR János, VADAI Gergely

Abstract. More and more of today's devices are electronic and software operated. Since they measure the signals of the real world and act as a result of processing, informatics is getting closer to interdisciplinary fields. The technology of everyday devices, automotive industry, Industry 4.0, Internet of Things (IoT) are based also on engineering, physics, electronics, biology and other fields besides informatics. Accordingly, the interdisciplinarity is important in education too, it is essential to teach related solutions of technology for the informatics teacher students since they will certainly need it during their teaching practice. Although the development of technology is still very rapid, the main principles of operation remain the same, so education should focus on this rather than on the teaching of a current software development environment or hardware realization. Following this idea, we have worked out several laboratory exercises for our programme of informatics teacher students. During their work they construct some circuits, write the code and practice the fundamental methods of embedded software development. We report on how they could understand and learn the most important principles during working on an educational, a commercial, an industrial and a medical field related exercise. The associated experience-based learning helps systematic understanding and development of creativity. Based on our experience, students are getting more self-confident by the use of modern technical systems, which we consider particularly important.

Keywords*:* technical informatics, Arduino, interdisciplinary education

1. Introduction

One of the most important issues of education is how to adapt to the rapid development of technology. Besides the experience and knowledge related to modern tools the students live in a rapidly changing environment, they used to get huge amount of information very quickly and accordingly their attitude can also be very different than it was earlier. It is a serious problem that pupils and students feel strange in the school environment; they do not understand why to learn several parts of the curriculum. This is getting to be a common approach in public thinking too: the school should teach what is needed in practice. It is a frequent debate if it is really needed to teach more universal knowledge today rather than teaching today's solutions and the use of tools directly. Is it necessary to do elementary math calculations without any tools, to know and to be able to prove the Pythagorean theorem? Is it required to teach physics, biology, and other science subjects as a separate subject?

In the field of informatics, the problem is particularly evident: we do not know what kind of tools, programming environments will be used at an IT company a few years later. So, it seems to be the best for students to acquire such knowledge that can help them to adapt to the quickly changing environment in the future. It is important to learn logical thinking and to understand the most important operating principles as they change the least. As IT solutions are getting common in more and more fields (education, industry, medicine, communication, navigation, entertainment, etc.), a certain level of interdisciplinary knowledge and openness is very useful.

Many people find it hopeless to understand the operating principles of modern devices, they are considered as "black boxes" even more than before, it is thought we can't see the details of operation. The devices often apply the latest results of technology; their operation is related to different fields of expertise. However, if we try to find the underlying principles instead of the details of the solutions, we can get a generalization that is understandable to many, the tools are often based on

the principles that can be observed in everyday life. It is supported by the various universal building blocks based on modern technology developed especially for education [1, 2, 3, 4]. They allow students to make rather serious tools by playing and enjoying the work. Therefore, it is important in teacher education to acquire in-depth knowledge and right attitude. The high-level and extensive knowledge of teachers is essential, what can't be replaced by the knowledge of modern educational methods, it can only be made more efficient by these. Our research group [5] has developed its activities according to the above. In order to support the education of informatics teacher students we develop courses, educational materials, tools and methods. We focus on practical education, and we aim to teach the required theoretical background and universal principles that can be learnt more efficiently during laboratory practices.

2. Technical methods and tools for informatics teacher training

The vast majority of today's devices are electronic, include processors, and are software operated. All the devices around us will be like this in the near future, regardless of size and application field. However, the construction and operation principles can be relatively simple and universal, as it is shown in Figure 1. Real-world signals can be translated to signals that can be handled by electronics and then these can be converted to numbers to allow digital signal processing. As a result, information is obtained, that can be used to act on the real world. Devices of this type are also called embedded systems due to the presence of the main operating unit, i.e. the built-in processor [6].

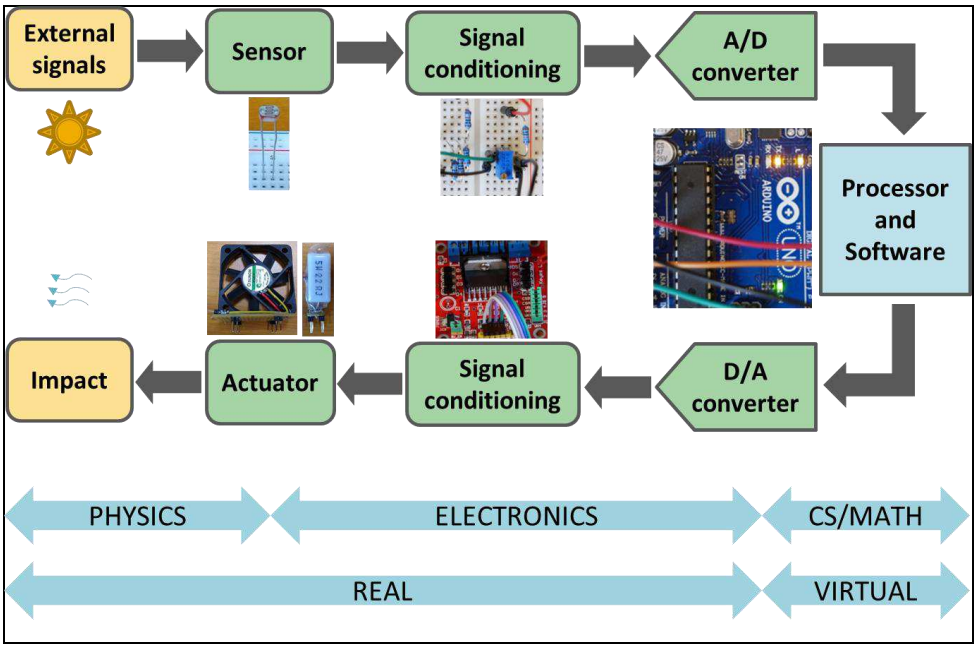


Figure 1. Block diagram of modern electronic devices.

This structure can be easily seen in many educational devices too, the teachers and students can make such systems using the available building blocks. They can design the algorithm of operation and write the corresponding code.

It is important to note that reliability is essential for all devices, embedded systems. Air bags must work in the right time, during patient monitoring software or hardware failure can risk life, but we also expect that mobile applications won't drive us in the wrong direction in a one-way street, that they make bank transfer to the right account and keep our data private. The catastrophe of a rocket is often mentioned as a typical example how a huge disaster can be caused by a simple inattention during the software code development [7]. In consequence, it is essential to develop a careful,

technology-oriented approach and attitude that directly appears in the official requirements of education output. It is instructive to see what standards apply to IT teachers in our country. Here we highlight a few things that are more directly related to the above:

- Encourages students to form their own opinions, helps to develop critical thinking, with particular emphasis on drawing the attention to the dangers of IT applications.
- Has the knowledge to enable learning and interpreting the new results in the IT field. Knows the basic research methodology of the field.
- Able to find and integrate the knowledge of different fields, especially mathematics and natural sciences.
- Able to apply the theoretical knowledge acquired in his specialty in practice, to convey it to the students.
- Is aware of the fact that knowledge and competencies developed in the field affect other fields as well.
- Able to use the tools of informatics education of the school professionally, to use these in teaching and distance education. Able to develop curriculum for informatics and to support the application of IT in the development of other curricula.
- Works with teachers of related fields. Able to coordinate the scheduling of teaching the subjects also present in related subjects.
- Committed to demanding teaching and continuous self-education.

The standards of required professional knowledge is extensive and includes the technical fields, robotics too.

2.1. Popular technical informatics education tools

Increasingly better, cheaper and more efficient tools are available that supports the learning about the architecture and operating principles of today's devices in a playful, enjoyable and practical way. Lego robots [1], single-board computers like, micro:bit [2], Arduino [3] and the Raspberry Pi [4] are good examples. They can handle many different sensors; several kinds of signals can be acquired. Students have direct access to the raw data what they can process, and they can produce impact as well accordingly. Therefore, the whole system is very transparent, the function of the sensors and other components can be clear even without knowing the details of the electronics background. Of course, there are many different levels of knowledge, but teachers must have in-depth education, they should not be confused, if the student ask, for example, how a robot can sense the nearby objects. Teaching of the course called "Technical applications of informatics" for the informatics teacher students at our university takes 15 and 30 hours per semester concerning the basics of theoretical background and practice, respectively. The curriculum includes robotics, application of the Raspberry PI, and particularly based on the Arduino platform, which is extremely popular worldwide.

2.2. Arduino applications – advantages and drawbacks

Arduino is development platform based on a single-board computer hardware and a simple user-friendly software development environment. It is very transparent, practically it is a microcontroller (a single chip integrating a CPU and peripherals) whose pins are connected to easy-to-use pin headers. Many additional components, sensors, actuators are available, therefore students can meet more directly with electronics and circuits. In addition to this, the developers also solved the programming with a very good sense of proportion. The programming language is C++, but in most

cases, it is sufficient to know the most general parts common in other languages as well. The environment is kept simple, there are a limited number of functions, but they are really useful. The architecture of the programs fits well to the application principles of the microcontrollers without using operating systems (setup and loop functions replace the main function). The microcontroller is a professional component of commercial, industrial and medical technology, essential in embedded systems, were reliability, following the well-established principles, standards and rules are all of particular importance. Of course, all of these can't be expected, but careless, improper use and development of undesired attitude should be avoided in any case.

There are a huge number of solutions for almost any task and problem on the internet. A considerable part of these are developed by hobbyists, students, so not by experts. There are many smart ideas, but sometimes the drawbacks are substantial especially regarding high quality education. On one hand, it can motivate the students to reproduce instead of working out own version, in most cases they don't even understand why the teacher asks them to solve a problem if there are available solutions of the internet. On the other hand, due to the lack of comprehensive expertise of hobbyists, there is a large number of virtually working but technically incorrect solutions. The numerous hits for a single problem can make it very hard to find the right one even for an experienced teacher. Unfortunately, even the official Arduino pages and the textbooks show sometimes inadequate guidelines what can cause hard-to-change habits. Therefore, we consider it important to help understanding the operation and principles as much as possible, to show the professionally correct applications, to develop self-reliance and care. We do not aim to provide step-by-step instructions, rather we show cases studies as examples to demonstrate the application of the right methods and approach. During the laboratory practicing the use of embedded software and hardware development methods mean experimental learning, help to be confident when problems have to be solved. In accordance with the requirements of education output, cautious thinking and right attitude can be improved, including careful consideration of the problem, ability to find a right solution, proper application of the professional methods, testing, finding and fixing the problems efficiently.

3. Arduino exercises

Laboratory exercises for the informatics teacher students are related to several different fields. Students work with various sensors, circuits, mechanical parts and see several methods and approaches. They acquire fire and working safety knowledge, they learn to record the phases of the work and practice collaboration. The lecturer evaluates and scores their work. The students are free to select a solution for the given problem; they can use any of the supporting material available on the internet. The selected problems effectively develop their problem-solving skills. First, they can give a simpler solution, then they are asked to develop a more advanced and more professional version. In order to support this the lecturer can give some guidelines and explanations. It is important to note that they experience how to work in a laboratory environment equipped with instruments and other hardware. They learn that there are important rules and standards in order to guarantee the required quality and reliability even if the reason is not necessarily clear at first sight. In the course we have used the following exercise examples that demonstrate how solutions need some knowledge of mathematics, physics and occasionally some other disciplines, what can be somewhat surprising for the students.

3.1. Measurement of voltage and resistance

Students certainly meet the problem of voltage measurement in most Arduino applications related to analogue signals. This is aided by the analogue-to-digital converter (ADC) of the microcontroller. Students learns the principle of converting a real value to integer number, they experience the effects of quantization and sampling, they understand the role of the voltage reference. The voltage can be expressed as [8]:

$$V_{in} = x \cdot \frac{V_{ref}}{2^b}, \qquad (1)$$

where x is the integer code at the output of the ADC, V_{ref} is the reference voltage (its default nominal value is 5 V for Arduinon Uno), b is the number of bits of the output code (10 for Arduino Uno, so the voltage resolution is $V_{ref}/1024$). The task was to build a simple voltage divider circuit on a breadboard, whose output was connected to one of the analogue inputs of the Arduino board. The students had to write a code to sample and calculate the voltage using equation 1.

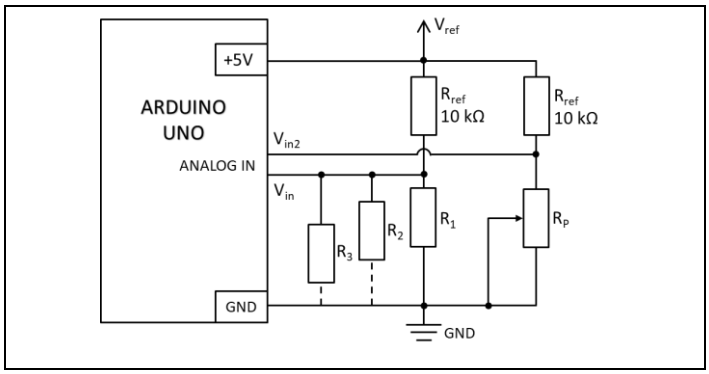


Figure 2. Voltage divider circuit connected to the Arduino board. One can select a resistor to measure any of the R_1, R_2 and R_3 by connect its lower end to the GND.

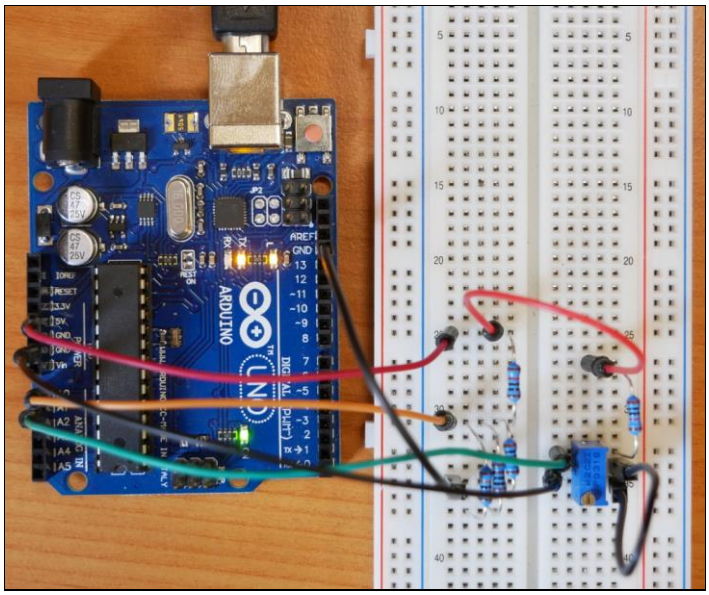


Figure 3. Voltage divider circuit assembled on a breadboard.

The Arduino integrated development environment (IDE) provides a so-called Serial monitor tool that can display the data sent by the microcontroller using only a few simple functions of the Serial library [9]. This is a quick method to see how the system is working and to find the possible errors.

The next task was to measure resistance. The voltmeter code had to be modified, but the students had to know and apply Ohm's law properly. The value of the resistor connected to ground (GND) can be obtained by measuring the voltage on it [10]. The same current flows through both resistors, its value can be easily obtained as V_{ref}-V_{in}/R_{ref}. Finally, the unknown resistance is given as the voltage on it divided by the current:

$$R = \frac{R_{ref} \cdot V_{in}}{V_{ref} - V_{in}}, \qquad (2)$$

where R_{ref} is a known reference resistor, V_{ref} is the reference voltage, V_{in} is the measured voltage. During the exercise the students had to measure the value of three resistors and to display the value of the potentiometer in real time. They got help where to find the Arduino library functions [9] required to start a conversion and to read the ADC code and to send the data to the host computer over the serial port. Student were facing the problem of integer division, where undesired rounding occurred and caused large error in the measurement. They have solved the problem by using floating point constant for numbers that could be represented also be integers (e.g. 5 V reference voltage). Some time was needed to find the reason of this error, but the success was a good experience for them. Being informatics teacher students, they were a bit less familiar with the electronics part, but they understood the main principles after completing the exercise.

3.2. Gate controlling system

Exercises that model the operation of an ordinary system are particularly useful. This directly demonstrates practical usefulness and the underlying principles. A related task was to assemble a model of a garage door actuator system and to write the operating code.

Moving a stepper motor had to be realized by working out the appropriate algorithm based on the knowledge of the operation principle and driving methods. The rotation of the gate between 0 and 90 degrees were provided by the stepper motor. This is also a good example of the possibility of theoretical abstraction, since in reality not a stepper is used for the purpose, but it only means the management of a component, the main principles are not affected. It was also a part of the task to implement an emergency stop using an optical sensor. If an obstacle was detected, the movement had to be stopped. In order to make a photogate they got a photocell whose resistance had to be measured. They built the circuit and wrote the corresponding code. Opening of the gate was started by pressing a button connected to one of the pins of the Arduino board configured as digital input.

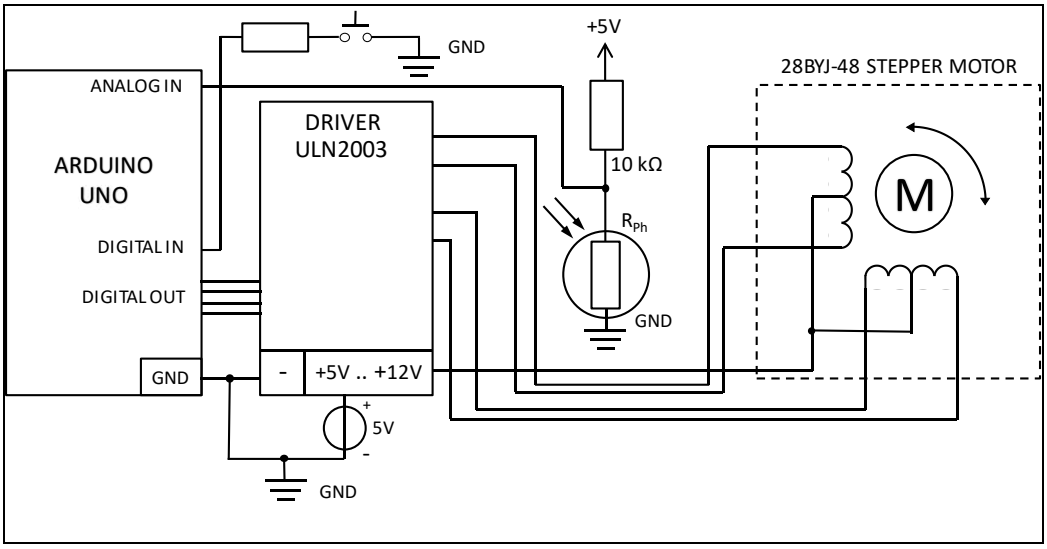


Figure 4. Arrangement of the gate controller system. The Arduino board controls a stepper motor using driver circuit. The gate opens upon pressing a button and the movement is suspended if the photogate detects an obstacle.

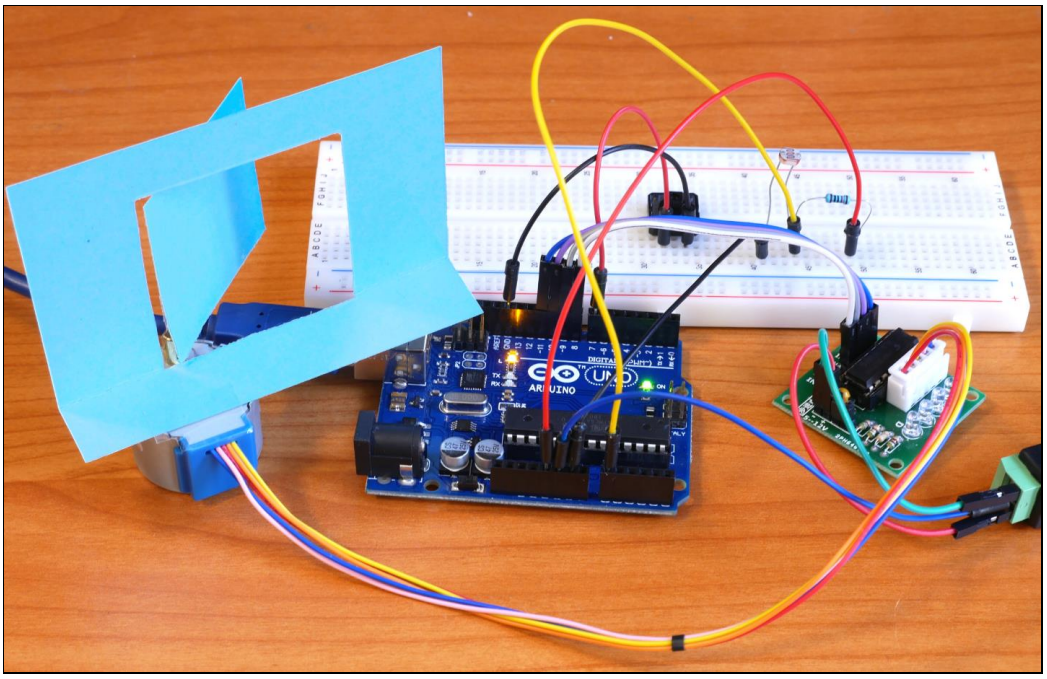


Figure 5. The realized gate controller model.

The task provides various options for solving the operation of the individual components. Half or full step can be used to control the motor, the speed of rotation can be changed, end positions can be detected. The photoresistor can be replaced by a phototransistor; infrared light can be used instead of visible light. The opening process can be controlled remotely; the operating logic can be varied. The movement speed can be reduced when approaching the end positions, a flashing LED can indicate that the gate is moving. The students usually write a for loop to do the required number of steps, however, the loop function can also be used with properly implemented conditional execution. Timing of the stepping can be solved more precisely using timer peripheral and interrupt techniques. These possibilities provide various levels of complexity to accommodate to students with different skills and experience. According to our observations the students had the following main difficulties: to understand the gearing of the stepper motor; to suspend the movement before

starting and during the presence of an obstacle. The mechanical construction and driving principle of the stepper motor was new to the students. The microcontroller code runs without an operating system, which was also unusual. Some more tutorials and simulations [11] can aid the understanding of the operation principles of the stepper motors. Examples and more detailed explanation of the loop function can also be useful to help straightforward implementation of suspending the motion.

3.3 Heart rate measurement using photoplethysmography

Today's smartphones and smart watches are capable of heart rate measurements by using one of the suitable principles, the so-called photoplethysmography. An exciting exercise can be built up to measure heart activity by an Arduino board too.

The principle of the measurement is rather simple: direct an infrared light into the finger and measure the intensity of the reflected or passing through light. This depends on the instantaneous blood volume in the finger [12, 13]. Therefore, one can get a time dependent signal mainly proportional to the blood pressure changes, so it can be used to detect the heart beats. The changing components of the signal is very small; it is superimposed on a large mean (DC) value. The students learn why and how to remove the large DC component and what are the simple methods to amplify the signal to fit into the input range of the ADC. The principle is rather general, it is applied to handle the signals of different sensors, e.g. a microphone. Building the circuit is a bit more what can be expected from the student during the lecture therefore in order to support the experimental work we provided our EDAQuino "shield" (plug-in board) that incorporates all of the required sensors and circuits [14, 15, 16]. Note that if longer time is available, students can build the circuit on a breadboard, so they can learn and experience more about electronic signal processing. Details can be found in our recent related article [17].

The students had to implement periodic sampling of the time dependent signal. This can be implemented in a simple and common way, however one should know the limitations of the method [17]. The Arduino IDE provides a simple chart recorder feature with the so-called Serial plotter tool to display the signal in real time. This allows the students to have an early success by visualizing their own heart function and they will be motivated to continue with the more complicated tasks too. Real-time heartbeat detection can be implemented in various ways, and a LED can be toggled to indicate when a heartbeat occurs. The task can be extended considerably, if the heartbeat time instants are available. Several heart rate variability indicators used in the medical practice can be computed including the mean (mean RR) and standard deviation (sdRR) of the beat-to-beat intervals (RR intervals) and the proportion of the intervals longer than the previous one by more than 50 ms (pNN50) [18, 19, 19]. Histogram of the RR intervals can also be calculated and displayed.

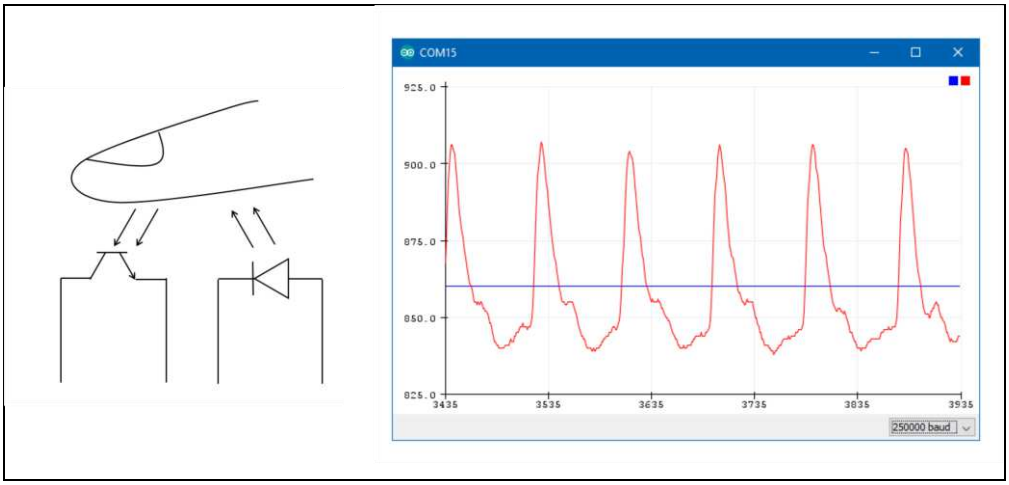


Figure 6. On the left-hand side, the arrangement of the finger photoplethysmography is shown. The plot on the right-hand side shows the Arduino's Serial plotter display of the phototransistors' output signal.

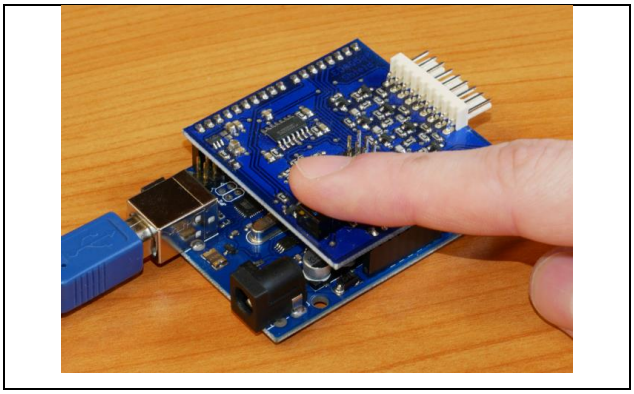


Figure 7. The students used the EDAQuino shield as a photoplethysmograph device.

We have found that designing and implementing the level crossing detection algorithm was not so easy for the students. This was mainly due to the lack of experience in real-time digital processing and the special programming requirements of the microcontroller were still not easy to follow. This highlights what could be the part of related education and what could be practiced during the teaching of different courses including introductory subjects of programming.

3.4 Temperature regulation

One of the simplest and most common method to keep something at the desired level is the on-off control applied in many everyday devices and systems. According to the general method of regulation, the deviation from the desired value has to be measured and increasing or decreasing should be forced accordingly. For on-off control, the strength of the effect is given, it does not dependent on the magnitude of the deviation from the desired value. In order to implement the on-off control, students used a power resistor that can be energized by applying voltage on it. The heating power depends on the voltage as follows:

$$P = \frac{V^2}{R}, \qquad (3)$$

where V is the heating voltage, R is the value of the power resistor that acts as a heating element.

This needs considerable current, that can be provided by external power circuits driven by some of the digital outputs of the Arduino board. A cooling can be solved by the use of a small fan or just by switching the heating off. A thermistor [21] (temperature dependent resistor) was used as temperature sensor. By measuring the resistance of this sensor, the temperature can be calculated using the following formula:

$$T = \frac{1}{\frac{1}{T_{25}} + \frac{1}{B} \cdot \ln\left(\frac{R}{R_{25}}\right)}, \qquad (4)$$

where T is the temperature expressed in Kelvins, T_{25} is the room temperature (25 degrees centigrade) in Kelvins (298.15 K), B is a material constant of the thermistor, R_{25} is the resistance of the thermistor at the T_{25} room temperature. The resistance measurement was already discussed in Section 3.1. Following that the resistance of the thermistor can be expressed as:

$$R_t = R_{ref} \frac{V}{V_{ref} - V}. \qquad (5)$$

The task was to write a program that heats up the power resistor to a given temperature, and this temperature had to be kept. Students had to use two levels to switch the heating and cooling on and off, and they had to monitor the temperature by sending the data to the Serial plotter in real time. They could observe the effect of the levels on the accuracy of regulation. The code can be easily changed to implement pulse width modulation (PWM) driving that effectively allows different levels of impact using a two-state signal. The experiment can be realized in other systems including charging a capacitor to a certain level via a resistor. In this case only a capacitor and a resistor is required provided that the resistor value is high enough to limit the charging current to a safe level [22].

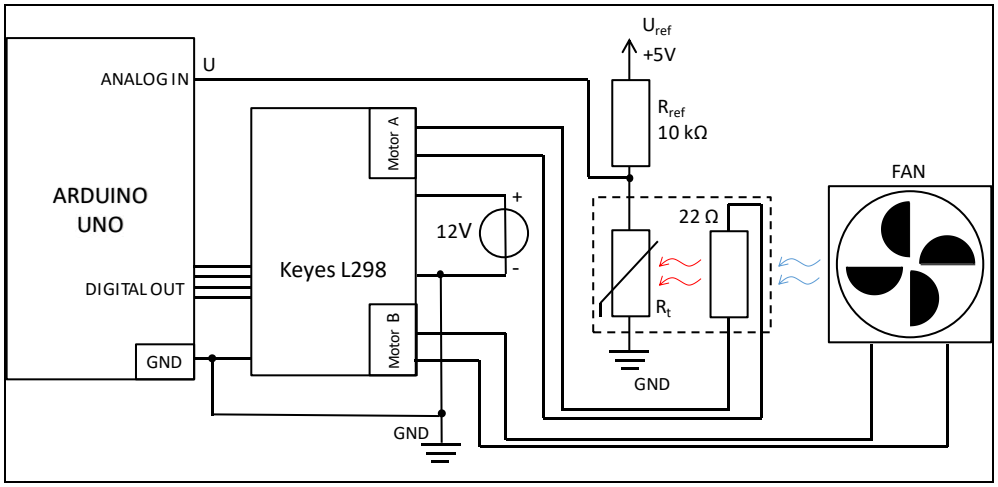


Figure 8. Block diagram of the system used to practicing on-off control. The Arduino board can switch on and off the heating and cooling via a higher power driver circuit (L298).

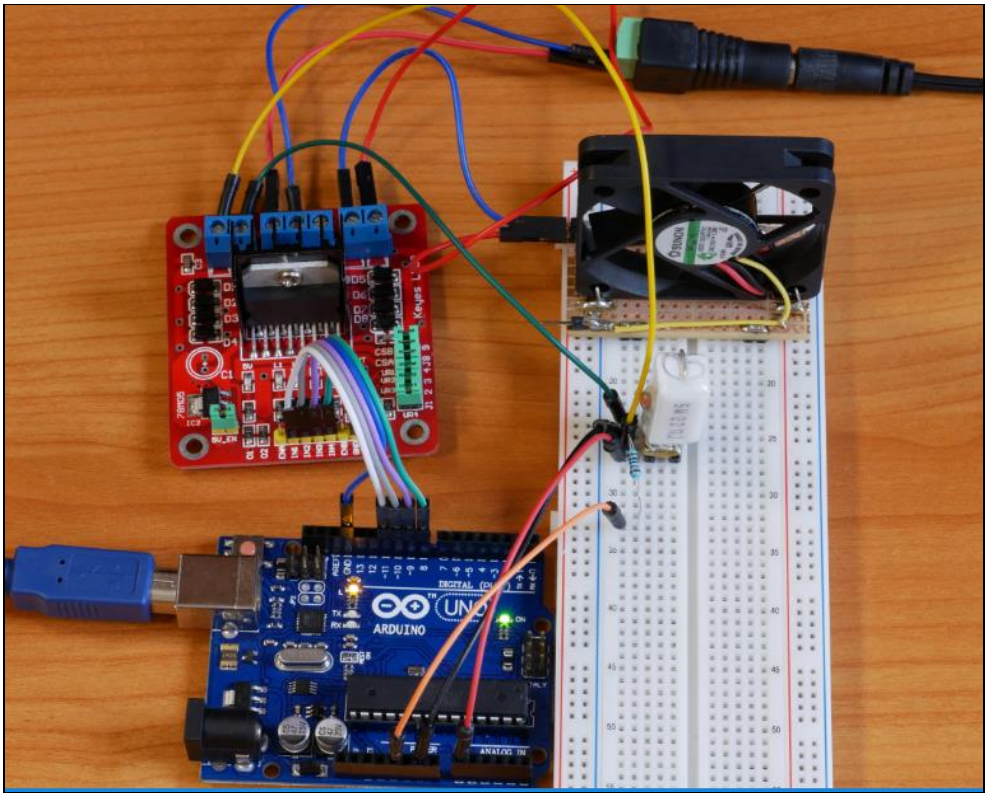


Figure 9. The assembled temperature control system.

We have experienced that one problem was to implement the hysteresis properly, and it was even harder to understand why the signal can be out of the levels for a short time and shown in Figure 10. Some everyday examples may help to understand the phenomenon better.

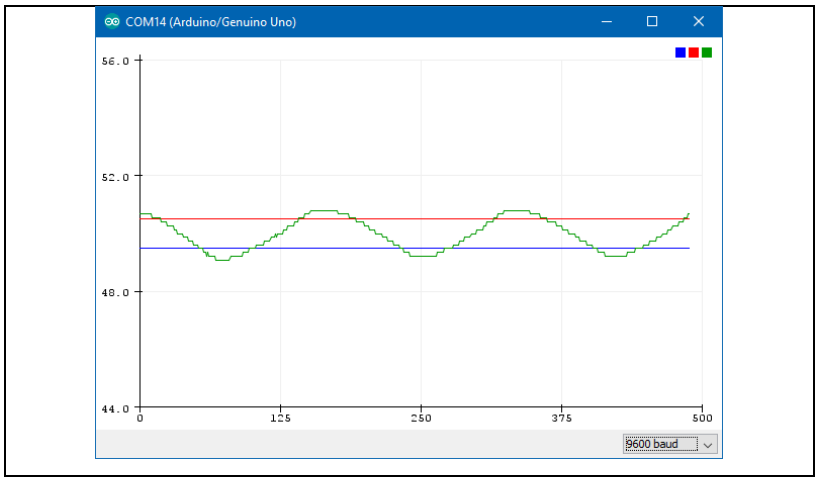


Figure 10. Temperature dependence during regulation (green curve). The levels of switching the heating and cooling on and off were 49.5 °C and 50.5 °C (blue and red lines).

During the implementation of Equation 4. sometimes integer division caused a problem, since in C and C++ the result of dividing two integers is an integer too, and it can result large errors. Emphasis should be placed on teaching the most important principles of programming embedded microcontrollers to avoid introducing errors during coding [23]. Note also, that this was the first course for the students to meet simple electronics, making the circuits using breadboards.

4. Conclusion

An important part of the training of informatics teacher students is to teach about the operation principles of the common electronic devices operated by processors and software. IT and other technical solutions are gaining importance in other areas too, so interdisciplinary nature is becoming widespread. Due to the rapid development of tools and software, it is essential to become more familiar with the universal principles and creative application of these. This can be effectively supported by laboratory practices in addition to theoretical education. Students can become more confident and demanding by gaining practical experience in a wider range, their problem-solving skills can be developed significantly.

We have shown four laboratory exercises for the Arduino platform used in the training of informatics teacher students. According to our experience, they could complete the tasks and found these interesting and exciting. The exercises can be expanded and used both in high school and university environments at various levels of complexity. Arduino circuits and accessories are readily available at a low cost, so classrooms can be equipped easily, a separate kit can be provided for every student. Students can use the same tools at home, which helps to be prepared and to realize their own ideas as well.

Acknowledgments

This study was funded by the Content Pedagogy Research Program of the Hungarian Academy of Sciences.

References

1. Information pages of Lego robots
https://www.lego.com/en-us/mindstorms , (last viewed: 29.04.2019)
2. Information pages of the Micro:bit
https://microbit.org/hu/, (last viewed: 29.04.2019)
3. Information pages of the Arduino
https://www.arduino.cc/, (last viewed: 29.04.2019)
4. Information pages of the Raspberry Pi
https://www.raspberrypi.org/ , (last viewed: 29.04.2019)
5. Homepage of the HAS Research Group of Technical Informatics Methodology
http://www.inf.u-szeged.hu/miszak/ (last viewed: 29.04.2019)
6. Embedded systems
https://en.wikipedia.org/wiki/Embedded_system, (last viewed: 29.04.2019)
7. Rocket disaster
https://hownot2code.com/2016/09/02/a-space-error-370-million-for-an-integer-overflow/, (last viewed: 29.04.2019)
8. Zoltán Gingl, Gergely Makan, János Mellár, *Arduino data conversion –1023 or 1024?* , (2018)
DOI:10.6084/m9.figshare.7434074.v1
9. Arduino functions
https://www.arduino.cc/reference/en/, (last viewed: 29.04.2019)

10. Gergely Makan, Róbert Mingesz, Zoltán Gingl, *How accurate is an Arduino Ohmmeter?*, (2019) Phys. Educ. 54 033001., DOI:10.1088/1361-6552/ab0910

11. Stepper motor https://en.wikipedia.org/wiki/Stepper_motor, (last viewed: 29.04.2019)

12. Nagy, Tamás, and Zoltán Gingl., *Low-cost photoplethysmograph solutions using the Raspberry Pi*, Computational Intelligence and Informatics, CINTI 2013, IEEE 14th International Symposium on. IEEE Budapest, Hungary (2013), pp. 163-167., DOI:10.1109/CINTI.2013.6705184

13. Zoltán Gingl, *A photoplethysmograph experiment for microcontroller labs*, INTERNATIONAL JOURNAL OF ELECTRICAL ENGINEERING EDUCATION (2012) 49 pp. 42-60., DOI:10.7227/IJEEE.49.1.4

14. Katalin Kopasz, Péter Makra and Zoltán Gingl, *Edaq530: a transparent, open-end and open-source measurement solution in natural science education*, Eur. J. Phys., (2011), 32 491, DOI:10.1088/0143-0807/32/2/020

15. Zoltán Gingl, János Mellár, Tamás Szépe, Gergely Makan, Róbert Mingesz, Gergely Vadai, Katalin Kopasz, *Universal Arduino-based experimenting system to support teaching of natural sciences*, GIREP-MPTL 2018 - Research and Innovation in Physics education:two sides of the same coin. 9th-13th July 2018, Donostia-San Sebastian, Spain (2018), DOI:10.6084/m9.figshare.6712541

16. Zoltan Gingl, Janos Mellar, Tamas Szepe, Gergely Makan, Robert Mingesz, Gergely Vadai, Katalin Kopasz, *Universal Arduino-based experimenting system to support teaching of natural sciences*, (2019) https://arxiv.org/abs/1901.03810; DOI: 10.1088/1742-6596/1287/1/012052

17. Zoltan Gingl, Gergely Makan, János Zsolt Mellár, Gergely Vadai, Robert Mingesz, *Phonocardiography, photoplethysmography with simple Arduino setups to support interdisciplinary STEM education*, (2018) DOI:10.6084/m9.figshare.7308356.v1

18. McKinley P S, Shapiro P A, Bagiella E, Myers M M, Meersman R E D, Grant I and Sloan R P, *Deriving heart period variability from blood pressure waveforms, Journal of Applied Physiology* (2003) 95 pp.1431–1438, DOI:10.1152/japplphysiol.01110.2002

19. Acharya U R, Joseph K P, Kannathal N, Lim C M and Suri J S, *Heart rate variability: a review*, Med Bio Eng Comput (2006) 44 pp.1031–1051, DOI:10.1007/s11517-006-0119-0

20. Tamás Nagy, Gergely Vadai and Zoltán Gingl, *Digital phonocardiographic experiments and signal processing in multidisciplinary fields of university education*, (2017) Eur. J. Phys. 38 055802, DOI:10.1088/1361-6404/aa7ae6

21. Thermistor https://en.wikipedia.org/wiki/Thermistor, (last viewed: 29.04.2019)

22. Zoltan Gingl, Robert Mingesz, Gergely Makan and Janos Mellar, *Driving with an Arduino? Keep the lane!*, (2019) Phys. Educ. 54 025010, DOI:10.1088/1361-6552/aafa41

23. Embedded C coding standards https://barrgroup.com/Embedded-Systems/Books/Embedded-C-Coding-Standard, (last viewed: 29.04.2019)

Authors

University of Szeged, Department of Technical Informatics, Hungary,
e-mail: makan@inf.u-szeged.hu

MAKAN Gergely

ANTAL Dóra

University of Szeged, Department of Technical Informatics, Hungary,
e-mail: antal74dora@gmail.com

MINGESZ Róbert

University of Szeged, Department of Technical Informatics, Hungary,
e-mail: mingesz@inf.u-szeged.hu

GINGL Zoltán

University of Szeged, Department of Technical Informatics, Hungary,
e-mail: gingl@inf.u-szeged.hu

MELLÁR János

University of Szeged, Department of Technical Informatics, Hungary,
e-mail: mellar@inf.u-szeged.hu

VADAI Gergely
University of Szeged, Department of Technical Informatics, Hungary,
e-mail: vadaig@inf.u-szeged.hu

KOPASZ Katalin

University of Szeged, Department of Optics and Quantum Electronics, Hungary,
e-mail: kopaszka@titan.physx.u-szeged.hu

About this document

Published in:

CENTRAL-EUROPEAN JOURNAL OF NEW TECHNOLOGIES IN RESEARCH, EDUCATION AND PRACTICE

Volume 1, Number 1. 2019.

ISSN: 2676-9425 (online)

DOI:

10.36427/CEJNTREP.1.1.385

License

XML as an Educational Curriculum Presentation of a Case Study

MENYHÁRT László Gábor

Abstract. XML can be found in all areas of information technologies. However, there is no such prevalence in education. In this article, I present the result of my case study in which I checked how XML and related technologies appear in education. The research was on the Internet and it processed several Hungarian and almost the same number of foreign universities' curriculum in English and German languages.

Keywords: case study, education, curriculum, Internet, XML

1. Introduction

Computer users can be grouped so that there are end-users and professionals. End-users use the computers for Internet, e-mail, office works and playing. Those professionals are less who use computers even more developing, operations or they can use computers lower level.

End-users do not meet XML directly, but they do not know any other file format either. However, these files appear on their computers, namely XHTML file can be seen in browser and at this time an XML file is downloaded into their computer. When they save an Office file like with *docx*, *xlsx* or *odt*, *ods*, etc. file extensions, these compressed files contain a directory structure with XML files storing configuration and data. Latters' format is OASIS Open Document Format (ODF) [7], former is Microsoft's own Open specification based on ODF and Open XML (ECMA-376 and ISO/IEC-2950). [8, 9]

So, if users do not see XML format directly, it appears in low level in a lot of places like configuration, storing data and communication. It used in many places. Here are some examples:

1. Configuration of webservices (domain.xml, …)
2. Configuration of applications (web.xml, …)
3. Configuration of operations like logging (log4j.xml, …)
4. Storing data in applications like Hungarian national tax and customs' general form filter application (Általános nyomtatványkitöltő program (ÁNYK)) uses *enyk* files with XML snytax, or UML diagrams are also XML files after saving, these are XML Metadata Interchange (XMI) files [10]
5. Data transfer with web-services (standard WS with SOAP)
6. Flexible using between versions, structure of XML allows to access part of data and ignore others.

So, XML is a good tool for integration between different systems.

I might highlight that big database systems like Oracle and DB2 types were developed to manage XML and special SQL syntax is available for filtering data with XML type. [11, 12, 13, 14]

At the same time development of native XML databases is started. [15, 16, 17] There were more experiments to implement one but only few stayed alive. Sedna's [18] last version (3.5) was published in November 2011, eXistDB' [19] version 3.0 can be downloaded from July 2015, Apache stopped development of xIndice in August 2011, after 10 years. Oracle bought Berkeley DB and its version 6.0.17 is available (though its documentation was updated in December 2009 for version 2.5.16 [20, 21]).

2. The case study

As we saw in the introduction, XML can be found in all part of informatics. But it has not so wide penetration in education.

In a case study the executor does the next steps: asks questions, selects the cases, decides about the technic of the data collection and analysis, collects data, evaluates and analyses them finally creates report. Case study is linear; it does not give changing proposal for handling possible problems founded at analysis of collected data. It gives only report. [23]

In this article I present the information about XML curriculum found in the research of education organizations. Results of my data collection in raw is available on link [1]. I explain in text here what I found when I researched where and how XML is taught. References are available in appendix in detail, so these are not mentioned here.

Method of data collection was online research, so it contains that information that is published via Internet, available and found by me. Google was used as starting point, at first free text search was used, later thematic search was run on sites of universities and filtering the pages. List of links contains every starting page. When searching in databases of the sites was available, I continued it there. Reading detailed descriptions and curriculums happened, too. I indicated the topic with "X" in the table in the appendix when it was mentioned exactly, and with "?" when it was not definitely written but affected most likely.

2.1. Hungarian institutions

At first, I investigated online published data of Hungarian universities.

XML Markup Language is available between programming languages on portal maintained **Eötvös Loránd University**'s (ELTE) Department of Programming Languages and Compilers. It appears on more courses like "Design, implement and manage databases" (IP-abATME/1), "Modern databases" (2+2) (IPM-08irKAEG/1) and "Theoretical foundations of information systems". Latter's thematic does not mention, only the available matters. Relevant topics are different and do not cover full palette. There is one more special collegium about XML, I will write about this in one of the following paragraphs.

There are more courses on **Budapest University of Technology and Economics** (BME) dealing with XML and related technologies. For example: XML-based system integration in business, XML and applications, The basics of XML. There are some between the university's normal courses, but some is started only on higher level vocational training. So, topics and quality of education can be different.

"XML programming" course was on **University Pannonia** Faculty of Information Technology in spring semester 2008/09. It taught XML, DTD, DOM processing and XSL transformation. Because of XSL materials should contain XPath, too. It seems that this course was available on spring semester 2011/12 for the last time. But course "Web programming" is available from 2008, where XML, XSD and XSL is part of the materials. XML is mentioned in courses „Electronic business technologies" and „Methods of modern system development". Bilicki Vilmos' materials about XML, XSD, DOM and XPath are for the course „Development of program systems". Holló Csaba's materials about XML, DTD and DOM are for course „Business web technologies".

Topics of "Database based systems" course on **University of Szeged** contains XML, DTD and native XML databases. I found only the first two in a lecture notes, but it does not mean that native XML databases were missed. Maybe there is another independent lecture note and it is about

XQuery, as well. XML is mentioned in courses "Application development", "Information technologies", "Web content development", "Business web technologies" and "Advanced programming II.". Bilicki Vilmos' course "Development of program systems" is available here, too.

XML can be found in a lot of description of courses on **University of Debrecen** Faculty of Informatics. It is negotiated in course "Programming technologies". "HTML, XML" pays attention on XSD, XPath and XSL. Course "XML data management" presents XSL, XSL-FO and XQuery, as well. „Advanced database technologies" teaches XQuery, too. Jeszenszky Péter's „Advanced XML technologies" presents XSD, XPath and XSL.

XML is present in course "Software technology I." on **Corvinus University of Budapest**. DTD, XPath and XSL are presented in course „Network technologies II.". Course „Internet application development" contains more, it teaches the previous ones and XSD.

XML is in professional and examination requirements of tertiary qualifications of Web-programmer training on **Budapest Business School** (BGE) University of Applied Sciences.

On the websites of **University of Dunaújváros** XML can be found in materials of courses „PERL" and „Web programming". But course "Internet technologies", available on engineer informatics basic training and Web-programmer advanced vocational training, deals with DTD, XSD, XPath and XSL topics in detail.

Course "Data management with XML" on **University of Miskolc** Faculty of Mechanical Engineering and Informatics processes DTD, XSD, SAX and DOM, XPath, XSL and native XML databases, XQuery topics in detail.

On **University of Nyíregyháza** the course "XML" of program designer informatics presents XML, SAX, DOM and XSL, so XPath, too. In teacher training the course „Applied systems" mentions XML, XSD and translations, which is probably XSL. In higher level vocational training XML based technologies is mentioned at „Related professional practice", and some books with XML topic is listed as recommended literature to course „Internet tools and services".

XML is mentioned only in course "Programming web systems on server side" on **University of Sopron**'s Web-programmer and General administrator Higher level vocational training, but course „ Programming web systems on client side" mentions XML, DTD, XPath and XSL, too.

Óbudai University's course „Theory and practice of database management" teaches XML topics in detailed, like XML, DTD, XSD, XPath, XSL and XQuery. Another course „Web-based technologies" teaches XSL-FO instead of XQuery, and XSD is not mentioned.

Websites of **Neumann János University** – earlier Pallasz Athéné University – contain XML at courses „ Corporate Information Systems I." and „WEB-programming II.". Native XML databases are available on course „Databases II.". XML and native XML databases are mentioned in course „Visual programming".

XML, DTD, SAX, DOM, XPath and XSL are present on course "Basics of Language Technology" on **Pázmány Péter Catholic University**.

On **Széchenyi István University** course „WEB-technology 1" deals with XML and DTD.

Some XML training material is available on websites of ***Free Information Society*** and Bíró Szabolcs' curriculum "Text processing based on XML" can be found on ***Tankönyvtár***.

2.2. Foreign examples

I was able to process only English and German web pages from foreign universities. I chose some universities from the first two hundred – as standard – because of their high number.

There are more courses dealing with XML on **Massachusetts Institute of Technology**. Thematic of course „Special Problems in Architecture Studies" mentions only XML, but the downloaded materials contain DTD as well. Course „Biomedical Information Technology" teaches XML, native XML databases and XQuery. Course „6.893: Database Systems" teaches XML, native XML databases, SAX and DOM processing, too. Course „1.264J / ESD.264J Database, Internet, and Systems Integration Technologies" is the most accurate because it negotiates the previous ones and XPath and XSL.

I found two courses with XML in course list of **Harvard Computer Science**. Course „XML with Java, Java Servlet, and JSP" deals with XML, DTD, XSD, SAX, DOM, XPath, XSL and XQuery topics. Course „Web Development Using XML" teaches the previous ones and XSL-FO and native XML databases.

In the course list of **Stanford** I found four courses. Course „Callback Me Maybe: Contemporary Javascript (CS 42)" mentions XML and DOM processing. Course „XML Data" teaches XML, DTD and XSD. Course „XML and Databases (CS 345B)" last in 2007 negotiated the previous ones and XSL and XQuery. While course „Introduction to Databases (CS 145)" mentions XPath, too.

On website of **Berkeley** I did not found any course about XML in last years. But earlier between 2003 and 2005 there was a course „XML and Related Technologies" that taught XML, XSD, XPath and XSL topics. Between 2007 and 2011 course „Concepts in Computing with Data" negotiated R programming language for statistic calculations, and it mentioned XML. Between 2006 and 2013 course „XML Foundations" was about XML, XSD, XPath, XSL and XQuery.

On **Oxford** a course „Extensible Markup Language" was found that presents XML, XSD, XPath and XSL topics.

In the course list of **University of Toronto** I did not find any course that negotiate XML.

In the course list of **University of British Columbia** I found only one course „Introduction to Database Systems" between 2005 and 2016 which mentions XML, XPath and XQuery.

XML, DTD and XPath are present in material of courses „Using R for Data Analysis and Graphics" and „Programming with R for Reproducible Research" on Swiss **Eidgenössische Technische Hochschule** (Zürich). Probably SAX, DOM and XSL should be negotiated. Not only the materials of Department of Statistics about R programming language contain XML, DTD, XSD, XPath and XQuery, but these are present in materials of the course "Big Data" since 2012, as well. Unfortunately, other and newer courses are not available in public.

On websites of the Australian **Monash University** I found materials of the course „Information retrieval systems" between 2006 and 2008 with code CS3201. XML, DTD, XSD and XSL was present, but XPath must be taught as well. I was unable to check whether there were changes in the material since then because login is required, and code was changed to FIT5166.

The University of Sydney 's course „E-Business Engineering" in 2007 has material via Internet with code COMP5347. This contains XML and XSL, but probably XPath, DTD and XSD is touched. Material of course „Foundations: Internet Software Platforms" (ELEC5742) contain XML and DTD.

Websites of **National University of Singapore** mention XML, but unfortunately, I did not find any material via public Internet to prove the appearance in education.

From Germany I checked materials of **Freie Universität Berlin** at first, where the materials are public only before 2008. Thematic of course "Datenbanksysteme II" mentions XML and DTD, but the materials are about SAX, DOM, XPath, XSL and XSL-FO as well. Thematic of course "Softwaretechnik" does not mention any part of XML topic, but XML and DOM is present in the materials. The same happened with course "Projekt Webdienste Wong - World of Networked Games". XML, DTD, XPath and XSL are mentioned in course "Digitale Editionsmethoden".

On websites of **Technische Universität München** I found a lot of reference to XML and related technologies. Material titled "Transformation von XML-Dokumenten" contains XML, XPath, XQuery, XSL and XSL-FO topics. Course "Einsatz und Realisierung von Datenbanksystemen" contains a topic "XML und Datenbanksysteme" that teaches XML, XPath and XQuery-t. In course "Foundations in Data Engineering" the topic "Other Data Models" negotiates XML, DTD, XSD, XPath and XQuery. Course "Database-Supported XML Processors" in year 2007/08 presented in detailed the XML topics except XSL-FO and native XML databases.

According to **Universität Tübingen**'s websites course "Database-Supported XML Processors" was available with the same content and teacher mentioned in the previous paragraph till year 2013. From onwards course "Advanced SQL" negotiate less topics.

After the universities I looked for online courses, whether how XML is educated. Starting point was a list of online courses and collection of MOOCs. For example, ***coursetalk.com*** contains courses from KhanAcademy, Stanford Online, Coursera, edX and other portals. XML is mentioned here in 184 courses, and 19 are free. Seven courses are rated with 4 stars out of five, 4 of them are free.

On site ***open2study.com*** I did not find any courses dealing with XML at the time of the search.

XML was mentioned in two courses on site ***edX.org***. These are „Professional Android Developer" by GalileoX and „Supply Chain Technology and Systems" by MIT.

8 courses mention XML on ***Alison***, but they are not exactly about it.

Lynda's 6 courses mention XML.

On site ***Udemy*** there is a concrete course about XML and XSD: „XML and XML Schema Definition in Easy Steps", and course „Learn XML Programming" is about XPath and XSL topics, there are another 39 courses mentioned XML.

A course from site ***online-learning.com*** that lasts 6 weeks, needs 4 hours per week activity and costs $ 279 is the most detailed and teaches XML, DTD, XSD, XPath, XSL, XSL-FO and XQuery.

Education materials are offered by ***Association for Computing Machinery*** (ACM). These are published every few years. The „Computing Curricula 2001 Computer Science" published December 2001 did not mention XML, but the „MSIS 2006: MODEL CURRICULUM AND GUIDELINES FOR GRADUATE DEGREE PROGRAMS IN INFORMATION SYSTEMS" version from 2006 contains „IT Infrastructure (Level 2)" where the webservices deal with XML. The „Information Technology Curriculum Guidelines for Undergraduate Degree Programs in Information Technology" published in 2008 contains „Information Management (IM) Data Organization Architecture, Managing the Database Environment, Intersystem Communications, Data Mapping and Exchange" and „Web Systems & Technologies" topics where XML, DTD, XSD, SAX, DOM, XPath and XSL technologies are listed. At the same year another publication, „Computer Science Curriculum 2008" mentions XSD as well in topic „Information Management

DataModelling". The same publication from year 2013 has two other topics „Data Modeling" and „Introduction to Databases" where XML, DTD, XSD, XPath, XSL and XQuery are present, too.

3. Presentation of my course „Data handling – XML"

In the previous paragraph I wrote about where and how XML is taught. IN this section I present my "Data handling – XML" course available on ELTE as a special collegium with IKI-AXFG and TANM-INF-300-XML codes. I teach this with my own thematic and materials since spring semester 2002/03.

At first it starts as an optional block on teacher training program, but software designers signed up in 50-50%. The aim of the course is to recognize databases and their integrated environment. At that time on the teacher trainer program's students had 2+2 (later 2+3) lessons per week with databases, this timing is not enough to know databases in proper depth, because this knowledge is important nowadays. Because of the high prevalence of W3C standards and semantic WEB we selected the XML technology to show the students formal and formatted data. So, the students learned markup languages for a semester. They learned about possibilities of storing method in semi-structured databases. They learned about native XML databases and XML type extensions of relation model.

The condition of practical mark is to present an assignment in XML topic and write a test at least with adequate qualification.

Topic contains the followings: representation of databases, description of data, XML syntax and well-formed definition. Constraint data with document type definition (DTD), schema (XSD Schema) and validation. Programming possibilities like SAX and DOM. Addressing data members with Xpath. Transformation of XML document (XSL, XSLT), formatted appearance (even in PDF). Native XML databases, for example Xindice. The detailed thematic per weeks:

1. Representation of databases, description of data, syntax, introduction of XML based on HTML knowledge, with attention regard to syntax and well-formed property
2. Document Types Definition: its syntax and validation (DTD)
3. XML Schema syntax and validation (XSD)
4. Presentation of Simple API for XML and Document Object Model with Javascript or Java
5. Comparing SAX and DOM, using them in different cases
6. Query languages, addressing nodes of XML document. Syntax and functions. (XPath)
7. Different style definition possibilities, XML Schema language's grammar and functions (XSL)
8. Transformation on client and on server side with XML Schema Language (XSLT)
9. Presentation of a native XML database, Xindice. Installation, administration
10. Handling XML data in Xindice. Using query languages in practice, XPath examples
11. XML:DB, Usage of XUpdate
12. Application of XML document, usage with different technologies: JavaScript, Java, JSP, PHP
13. Test
14. Presentation of assignments, replacement of tests if necessary, administration

We specified [24], [25], [26] and [27] as recommended literature beyond the next links: [28], [29], [30], [31], [32], [33] and [34].

Structure of a 90 minutes lesson per week is the next: the first half part is an interactive presentation about the new knowledge, the second practise part is for solve the sample tasks.

The new technologies learned in this semester must be used and presented in the assignment task. The test stands in two parts. First half an hour is the theoretical part, when concepts, differences and relationships are written on paper. Next on the practise part looking for mistakes correction and create new functions are the tasks on computer int his topic. Link [2] contains an example test.

Students' feedback shows that they like the subject and have learned interesting and useful things. Some feedback is available on links [3], [4], [5] and [6].

4. Conclusion

My curriculum about XML and related technologies can be shown at the early stage of the current informatics training on universities and later students can build their learning knowledge on this.

Previous paragraphs show that XML can be found in a lot of universities' curriculum. Somewhere there is only some lessons about XML in a course, elsewhere there is a whole course about XML. In some places it is mentioned as a data representing model, but sometimes it is compared with structured databases.

Teaching XML is inescapable todays informatics training found with a lot of my colleges, because it is mentioned more times, compared with other technologies, and some is based on this. But XML is not present between the mandatory foundation courses.

Teaching another, „Web-development" course realized that XML could be good at an early stage on the university. So, courses could build onto this and could use the concepts and methods if there is an introductory course about XML.

My curriculum is designed for a semester and it is one of the most extensive study materials which present the structured data and types. It introduces the world of text files, sequence processing and tree data structure in the memory. It encourages abstract, higher-level thinking through configuration constraints.

References

1. *Materials collected for the case study* (2017) http://xml.inf.elte.hu/archive/konferencia/2017/INFODIDACT/gyujtes_v2.xlsx (last checked: 2019.06.30.)
2. *A semester test paper* (Hungarian) (2017) http://xml.inf.elte.hu/archive/konferencia/2017/INFODIDACT/AkXML_zh.jpg (last checked: 2019.06.30.)
3. *Opinion of student 1* (Hungarian) (2017) http://xml.inf.elte.hu/archive/konferencia/2017/INFODIDACT/AkXML_velemeny_1.jpg (last checked: 2019.06.30.)
4. *Opinion of student 2* (Hungarian) (2017) http://xml.inf.elte.hu/archive/konferencia/2017/INFODIDACT/AkXML_velemeny_2.jpg (last checked: 2019.06.30.)

5. *Opinion of student 3* (Hungarian) (2017) http://xml.inf.elte.hu/archive/konferencia/2017/INFODIDACT/AkXML_velemeny_3.jpg (last checked: 2019.06.30.)

6. *Opinion of student 4* (Hungarian) (2017) http://xml.inf.elte.hu/archive/konferencia/2017/INFODIDACT/AkXML_velemeny_4.jpg (last checked: 2019.06.30.)

7. *OASIS Open Document Format* (2011) http://docs.oasis-open.org/office/v1.2/cs01/OpenDocument-v1.2-cs01.pdf (last checked: 2019.06.30.)

8. *Microsoft Office Open XML Format* (2017) https://msdn.microsoft.com/en-us/library/gg548604(v=office.12).aspx (last checked: 2019.06.30.)

9. *Microsoft DOCX Format* (2015) https://docs.microsoft.com/en-us/openspecs/office_standards/ms-docx/b839fe1f-e1ca-4fa6-8c26-5954d0abbccd (last checked: 2019.06.30.)

10. *XML Metadata Interchange* (2016) http://en.wikipedia.org/wiki/XML_Metadata_Interchange (last checked: 2019.06.30.)

11. A. Chaudhri, R. Zicari, A. Rashid: *XML Data Management: Native XML and XML Enabled DataBase Systems*, Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA, 2003, ISBN:0201844524 http://dl.acm.org/citation.cfm?id=599754 (last checked: 2019.06.30.)

12. *Oracle White Paper* (2017) http://www.oracle.com/technetwork/database-features/xmldb/overview/xmldb-twp-12cr1-1964803.pdf (last checked: 2019.06.30.)

13. *Native XML support in DB2 universal database* (2005) http://dl.acm.org/citation.cfm?id=1083727 (last checked: 2019.06.30.)

14. *DB2* (2013) https://www.ibm.com/developerworks/community/wikis/form/anonymous/api/wiki/bb5007a3-d52b-44e4-a2bc-4e56b454ff0f/page/5ac75e56-a02f-425d-91a5-ab0719e65416/attachment/ef7b51a7-9568-4fa4-82c0-10a8a1d35726/media/replacing_xml_extender.pdf (last checked: 2019.06.30.)

15. H. V. Jagadish, S. Al-Khalifa, A. Chapman, L. V. S. Lakshmanan, A. Nierman, S. Paparizos, J. M. Patel, D. Srivastava, N. Wiwatwattana, Y. Wu, C. Yu: *Timber: A native XML database* In The VLDB Journal — The International Journal on Very Large Data Bases archive, Volume 11 Issue 4, December 2002, Pages 274-291 DOI: 10.1007/s00778-002-0081-x http://dl.acm.org/citation.cfm?id=764201 (last checked: 2019.06.30.)

16. *Native XML Databases* (2016) http://cs.ulb.ac.be/public/_media/teaching/infoh415/student_projects/xml_databases.pdf (last checked: 2019.06.30.)

17. *XML Database Products: Native XML Databases* (2010) http://www.rpbourret.com/xml/ProdsNative.htm (last checked: 2019.06.30.)

18. *Sedna* (2012) http://www.sedna.org/ (last checked: 2019.06.30.)

19. W. Meier: *eXist: An Open Source Native XML Database* In Web, Web-Services, and Database Systems, Volume 2593 of the series Lecture Notes in Computer Science pp 169-183 2003. DOI: 10.1007/3-540-36560-5_13, http://link.springer.com/chapter/10.1007/3-540-36560-5_13#page-1 (last checked: 2019.06.30.)

20. *Documentation of Berkeley DB* (2017) http://www.oracle.com/technetwork/database/database-technologies/berkeleydb/documentation/index.html (last checked: 2019.06.30.)

21. *Documentation of Berkeley DB XML* (2017) http://docs.oracle.com/cd/E17276_01/html/toc.htm (last checked: 2019.06.30.)

22. S. K. Soy: *The case study as a research method* (1997) https://www.ischool.utexas.edu/~ssoy/usesusers/l391d1b.htm (last checked: 2019.06.30.)

23. *Difference Between Action Research and Case Study* (2017) http://pediaa.com/difference-between-action-research-and-case-study/ (last checked: 2019.06.30.)

24. S. Abiteboul, P. Buneman, D. Suciu: *Data on the Web, From Relations to Semistructured Data and XML*. Morgan Kaufmann, San Francisco, California, 2000 DOI: 10.1002/1097-4571(2000)9999:9999%3C::aid-asi1016%3E3.0.co;2-z

25. N. Bradley: *The XML companion*, Addison-Wesley Professional, 1999 DOI: 10.1162/10996629952104395

26. M. J. Young: *XML step-by-step*, Microsoft Press, 2002

27. B. McLaughlin: *Java and XML*, O'Reilly Media 2001

28. *World Wide Web Consortium* (2017) http://www.w3.org/ (last checked: 2019.06.30.)

29. *W3C - Extensible Markup Language* (2016) http://www.w3.org/XML/ (last checked: 2019.06.30.)

30. *XML - W3C Recommendation* (2008) http://www.w3.org/TR/xml/ (last checked: 2019.06.30.)

31. *XML 1.0* (2008) http://www.w3.org/TR/2008/REC-xml-20081126/ (last checked: 2019.06.30.)

32. *Document Type Declaration* (2008) http://www.w3.org/TR/REC-xml/#dt-doctype (last checked: 2019.06.30.)

33. *XML Schema Definition Language* (2012) http://www.w3.org/TR/xmlschema11-1/ (last checked: 2019.06.30.)

34. *W3Schools Online Web Tutorials* (1999-2017) http://www.w3schools.com/ (last checked: 2019.06.30.)

Author

MENYHÁRT László Gábor

Eötvös Loránd University, Faculty of Informatics, Department of Media and Educational Informatics, Hungary,
e-mail: menyhart@inf.elte.hu

About this document

Published in:

CENTRAL-EUROPEAN JOURNAL OF NEW TECHNOLOGIES IN RESEARCH, EDUCATION AND PRACTICE

Volume 1, Number 1. 2019.

ISSN: 2676-9425 (online)

DOI:

10.36427/CEJNTREP.1.1.384

The Role of Input-Output Management in Programming Education

HORVÁTH Győző

Abstract. In the introductory programming education, during coding input and output management often suppresses in proportion the essential parts of the code. A novice programmer is essentially given three tasks with the solution of the original problem: reading the input, processing the data and writing the output. This article attempts to explore the role of input and output in the code, and in some cases, how to make them easier to implement.

Keywords: programming education, methodology, input-output

1. Introduction

Programming is basically data processing: the final result must be produced from the initial values. Computers are called to help in the process of this calculation. This process can be examined from different aspects. Programming education focuses on how computer-assisted problem solving can be done systematically. After reading and understanding the text of the task, the first three of the steps of problem solving are important to us: 1) define and describe in a formal way *what* the task is (*specification*), 2) describe in an abstract language *how* to solve the problem (*algorithm*), and 3) to *implement* the former solution of the task in a specific programming language (*coding*). Breaking the problem into steps is an important methodological support in achieving the basic goal of introductory programming education, where we expect the student to

- understand the task;
- recognize the input and output data in the text;
- describe the structure of those data;
- formulate the relationship between input and output data (either textually or formally);
- break down the solution into elementary steps;
- implement the solution in a specific programming language;
- check the correctness of the solution.

Students in this process need to learn a lot of new knowledge: the formal language of the specification (although it can be replaced by textual or visual descriptions in some circumstances), the algorithm description language (Nassi–Shneiderman diagram, aka NSD, sentence-like description, pseudocode) and the specific programming language itself. The purpose of this article is not to consider how much this is necessary, how to simplify or remove it, but rather to consider the above process as given. It is very important not to burden students with unnecessary information in this learning process.

The other aspect of looking at programming as data processing is considering how the initial data gets into this process and how the results come out from the process to the user. The first operation is *reading the input values*, and the second one is *writing the output values*, and together they are called *IO operations*.

The IO operations are very important at the early stage of learning programming. It allows interaction with the computer, which brings the bare bone hardware closer to the student by giving it human attributes. With the help of those operations the student can start a conversation with the computer even if it is programmatically designed in advance. This is a big methodological advantage in programming education. In addition, without output operation the user would not know the result of the calculation. So, at least, outputting data is inevitable in programming, and fortunately is quite simple in most programming languages.

On the other hand, reading input can be a cumbersome process. The starting point of this article is the observation, that it is often found during coding – especially in the early stages –, that reading the input and writing the output (but mainly the former) require disproportionally more codes and programming language knowledge than the solution itself. Usually, the reading operation itself is simple, but the communication with the user, filtering out the user's accidental or intentional errors, and checking the sometimes complicated relations of preconditions, often leads to unnecessarily lot of codes, and requires the use of language elements that should not be introduced in the initial phase of programming. Many times, concepts such as input buffer, compiler switches, type conversions need to be explained to the students, for which they are not prepared enough, or these concepts unnecessarily take time and effort from the lesson. The primary goal in programming education is not how to communicate safely, understandably and politely with the user (those are important, but secondary goals), but how to represent the data in the problem and how to generate the output data from the input data, i.e. how to solve the problem. Beside this, the way of obtaining the data (standard input, file, database, network query) and writing the end result (standard output, file, etc.) can be considered as a separate task. In other words, when students work on the coding of a task, they suddenly have to solve three tasks: besides the original task, they have to carry out the input and output parts as well.

It is interesting that the literature has very few references regarding to this topic. Input and output come to the foreground when it is mentioned among best educational practices that input values should always be validated [1], but does not mention the complexity coming with it. In an other article, which defines its own educational programming language, a new abstract interface is defined to hide the original complexity of input operations [2].

In this article, we look at the role of input and output handling in computer-assisted problem solving, how we can make them easier to let the students focus on the real task and investigate the role of the whole input-output management. The problem is investigated in the context of introductory programming education in higher education, that is why the code examples are in C++, but the concepts of this article may be applied to lower levels of programming education as well.

2. The problem

Reading the input in a program may be exciting, necessary, and on its own it is usually quite simple in most of the programming languages. However, many times the attention shifts towards the details of the reading process from the aim of this operation, i.e. getting the initial data for the problem-solving part. Tasks like *"check the fulfilment of the precondition"*, or *"check the type of the input"*, or *"exit the program if something fails"*, or *"repeat the reading until it is good"* should be considered different and separate tasks, which usually require advanced concepts of programming. Getting those tasks too early may cause confusions in beginner students.

The input may be given in a file which is read from the standard input. The structure of the file sometimes allows easy reading operations, sometimes it makes it harder. Typical examples for the latter are reading until the end of the file, or numbers and texts in the same line (Figure 1), or using an unusual separator in a line (Figure 2).

```
Some text here 1 2 3
```

Figure 1: Input line with mixed content

```
Some text;1;Other text;3
```

Figure 2: Special separator in input line

Handling input may take a lot of time, may result a lot of lines of code, and complex reading logic may introduce errors which makes the reading process unreliable for the main problem-solving part.

3. Example task

Consider the following task, which serves as a reference for the rest of the article!

Task: Two positive numbers are given, calculate the sum of those numbers!

```
Input:          a∈ℕ, b∈ℕ
Output:         c∈ℕ
Pre-condition:  a>0 and b>0
Post-condition: c=a+b
```

Figure 3: The specification of the example task

The algorithm in its simplest form (NSD or pseudocode):

```
c:=a+b
```

Figure 4: The short version of the algorithm

The pseudocode form is usually written as a separate subroutine:[1]

```
Type TInt=Integer
Procedure Task(Const a, b: TInt, Var c:TInt)
    c:=a+b
Procedure end
```

Figure 5: Longer version of the algorithm

The corresponding C++ code with the simplest reading of input is short as possible, but the reading part may result a lot of lines of code in more complex cases (see Appendix 1):

[1] The first line of the algorithm indicates that the simple and complex types in the specification can be defined before the procedure. Usually complex types are defined separately for later references, but there is nothing to prevent us from naming simple types. Producing the algorithm can thus become more schematic.

```cpp
#include <iostream>
using namespace std;
int main() {
    int a, b, c;          // declaration
    cin >> a >> b;        // reading input
    c = a + b;            // main process
    cout << c << endl;    // writing output
}
```

Figure 6: The C++ solution of the example task

4. Input and output in the specification and algorithm

The specification (Figure 3) does not have a *direct* impact on reading the input and writing the output. The input and output section describes the data and their structure required to solve the task. The pre-condition narrows the range of possible input data, and the post-condition specifies the connection between the input and output data. However, none of those parts deal with where and how the data comes from or goes. The specification deals with the conditions that must exist *before* and *after* solving a specific task (i.e. the main process).

As can be seen from the example algorithms (Figure 4 and Figure 5), the algorithm does not include, at least in the commonly used shorter form, reading and writing operations. At the same time, when constructing the algorithm, whether it is a NSD or a pseudocode, we have many implicit assumptions: 1) it is known from the specification which data are the inputs and outputs; 2) their types are also known from the specification; 3) the input data is correctly given at the beginning of the algorithm. Implicit assumptions become evident if we construct the algorithm of not only the essential task, but the whole program (Figure 7).[2] This is worth doing because it can show the general structure of the programs, where each program consists of three main parts: *reading input*, *processing* and *writing output*.

```
Const ...                    <-- see short algorithm
Type TInt=Integer            <-- see short algorithm
Var  a, b, c: TInt
Program
    In: a, b [a>0 and b>0]
    c:=a+b                   <-- see short algorithm
    Out: c
Program end
```

Figure 7: The algorithm of the whole program of the example task

In this version, our previous assumptions become explicit: it can be seen what was read and what was written, what the types of the variables are, and it is also clear that processing is started with correctly read input values, and the resulting data must be written out. However, usually it is not necessary to write such detailed algorithm, as 1) the name and type of variables can be "generated" from the specification, and 2) the structure of the program always task-independently builds up from the triple of input-processing-output, and finally 3) the input and output part also can be "generated" according to the specification. What remains is the essential part: the specification of the data structures (with type definitions, if necessary) and the sequence of elementary operations to solve the task.

With subroutines, the algorithm of the example task looks like this:

[2] The examples are given with pseudocode in the following.

```
Const ...                          <-- see short algorithm
Type TInt=Integer                  <-- see short algorithm
Var  a, b, c: TInt
Program
    Read(a,b)
    Process(a,b,c)
    Write(c)
Program end

Procedure Read(Var a,b:TInt)
    In: a [a>0]
    In: b [b>0]
Procedure end

Procedure Process(Const a,b:TInt, Vált c:TInt) <-- see short algorithm
    c:=a+b                                     <-- see short algorithm
Procedure end                                  <-- see short algorithm
Procedure Write(Const c:TInt)
    Out: c
Procedure end
```

Figure 8: The algorithm of the example task with subroutines

In this case, it is also obvious that, apart from the data structure descriptions and the processing algorithm, everything else can be derived mechanically from the specification, and therefore those parts are usually omitted. However, the advantage of this detailed description is that it gives a firm recommendation on the code structure.

The above algorithm can be developed further. If the procedures were replaced by functions, the main program would look like this:

```
Program
    (a,b) := Read()
    (c)   := Process(a,b)
    Write(c)
Program end
```

Figure 9: The algorithm of the example task with functions

The parenthesized data on the left side of the assignment indicates that in this case the reading function should return a complex data structure containing the read data, and the processing function should return all the output data.

Furthermore, our detailed algorithm can be generalized. If a program is considered as a function mapping the output data from the input data, this concept can be reflected in the data structures:

```
Const ...                          <-- task-specific
Type TInt = Integer                <-- task-specific
Type TInput  = Record(a,b:TInt)    <-- task-specific
Type TOutput = Record(c:TInt)      <-- task-specific
Var  in:  TInput
     out: TOutput
```

Figure 10: Input-output data structures of the example task

The algorithm of the main program will thus become task-independent:

```
Program                                    Program
    Read(in)                                   in:=Read()
    Process(in, out)          OR               out:=Process(in)
    Write(out)                                 Write(out)
Program end                                Program end
```

Figure 11: Task-independent structure of the main program

The functional solution can be simplified to a function composition:

```
Write(Process(Read()))    <===>    Write∘Process∘Read
```

Figure 12: The functional approach of the algorithm of the main program

For procedures, the three subroutines can look like this:

```
Procedure Read(Var in:TInput)
    In: in.a [a>0]
    In: in.b [b>0]
Procedure end

Procedure Process(Const in:TInput, Var out:TOutput)
    out.c:=in.a+in.b
Procedure end

Procedure Write(Const out:TOutput)
    Out: out.c
Procedure end
```

Figure 13: Subroutines for the generalized input-output data structures

To avoid the continuous `in/out` references in the `Process` procedure, we need to do the following:

```
Procedure Process(Const in:TInput, Var out:TOutput)
Var a,b,c:TInt
    a:=in.a             <-- "read"
    b:=in.b             <-- "read"
    c:=a+b              <-- "process"
    out.c:=c            <-- "write"
Procedure end
```

Figure 14: "Reading" and "writing" in the main process in the case of the generalized input-output data structures

In the end, the structure of the main program became task-independent, and the task-specificities were moved to each subroutine. It should be noted, that the `Process` subroutine repeats the structure of the previous main program, as it reflects the read-process-output triple. The main difference in this case is that the "read" and "write" operations have been decoupled from the outside world, and the work can be done with clean data structures. On algorithmic level there are not so many benefits coming from this approach, because there are not many differences between "`In: a`" and "`a:=in.a`", but the simplicity of the latter can be utilized during coding.

Finally note that, regardless of a particular implementation, the program is executed in the following steps: 1) reading the data so that it is available in a predefined structure; 2) the main process results in a predefined structured output data, which are 3) written out. An interesting feature of this process is that each step is determined by the structure of the data. Defining the *data structure* is a very important part of the problem solution, it requires a number of conscious, algorithmic and code decisions, so this should be considered as the fourth (more precisely the zero) task of a problem solution. While reading, processing, and writing data can be solved independently of each other, each of them depends on the chosen data structure (Figure 15). Thus, special attention should be paid to specification and thus to the description of the data structure.

Figure 15: The role of the data structure on the read-process-write triple

5. Input and output in the code

At the beginning of programming education, there are mainly smaller tasks that make students practice the essential elements of problem solving and programming language. As it was already mentioned in the introduction, during coding, disproportionally amount of code needs to be written for reading the input compared to solving the main task. In this chapter, we will discuss what opportunities we have to simplify inputting. These techniques may be useful not only at the beginning of programming, but also later, when we want to deal with as many tasks as possible and not with their inputs.

5.1. Omitting input

If reading the input takes unacceptable effort, *the simplest if it is omitted*. In this case the defined input data structure is pre-filled with baked-in values. The solution leaves freedom to define the data structure that fits the task, only the input part is skipped immediately focusing on the task solution.

```
int main() {
    int a, b, c;        // declaration
    a = 3;              // reading input
    b = 5;
    c = a + b;          // processing
    cout << c << endl;  // writing putput
}
```

Figure 16: The C++ solution of the example task with baked-in input values

The disadvantage of this solution is that the program is not general: the program needs to be changed every time for specifying new input data. However, at this stage, the goal is not necessarily to write a general program, but to ensure that the student can solve the task with the provided data. They can learn the generalization later, e.g. with independent input reading tasks. The testing process is not much different in the omitted and general case:[3]

- omitted: *new data* → (*compile* → *run*) → *check the result*
- general: (*run*) → *new data* → *check the result*

A further disadvantage of this solution is that it cannot be tested automatically in this form.

5.2. Providing helper functions

Reading the input is often complicated, because it can be done with very low-level operations, and beginners need to use complex language elements to manage and control the standard input-output, and to transform incoming data. However, these low-level operations can be hidden behind *higher-level functions*. Of course, this helper library needs to be provided to the students somehow, for example a properly configured programming environment [3,4] can take care of it. The helper functions can be freely analysed, expanded, and thus contribute to the learning of input and output management. A great advantage of this approach, furthermore, that the

[3] The steps in parentheses can usually be done in one action in programming environments.

definition of data structures remains in the student's hands. Here are some examples for using such functions:

```
bool positive(int p) {
    return p > 0;
}

read(a);            // reading input
read(a, "a = ", "Positive number must be given!", positive);
read_line(s);
read_line(s, ';');
write(a);           // writing output
write_line(s);
```

Figure 17: Reading with helper functions

where, for example, verified reading can look like this:

```
template<typename T>
void read(T& p, string message, string errormsg, bool f(T)) {
    bool good;
    do {
        clog << message;
        cin >> p;
        good = f(p);
        if (!good) { clog << errormsg << endl; }
    } while(!good);
}
```

Figure 18: An example for a higher-level helper function for reading input

5.3. Pre-written input reading

Another option for making input reading easier is that the student receives a *prepared template* in which the input and output logic has already been implemented and should only complete the processing part correctly. As the reading part already fills up a given data structure, the freedom of data description is missing in these solutions. On the one hand, this can be a disadvantage, as it does not exercise this ability, but it can also be an advantage, as it can be used to show good patterns for data description of given problems, and also to practice adaptability, where you need to understand and work with existing decisions (soft skill).

5.3.1. Without subroutines

In the first case, the code is not divided into subroutines, the processing part is empty from the quadruple of the declaration-reading-processing-writing. Such tasks require proper preparation and can be used well in automatic assessment environments. Depending on the environment, additional option may be to hide certain parts of the prepared code or make it read-only. If the whole code remains editable, any language element (e.g. function) can be used. If the access is restricted, it depends on the programming language what elements can be used or not, that is why it is important to show the entire context of the code:

```
#include <iostream>
using namespace std;
int main() {
    int a, b, c;                // declaration
    cin >> a >> b;              // reading input
    // USER CODE BEGINS
                                // processing
    // USER CODE ENDS
    cout << c << endl;          // writing output
}
```

Figure 19: Pre-written code template without subroutines for the example task

5.3.2. With subroutines

With subroutines, the basic concept of the previous sub-section remains valid, only the body of the processing function is not filled (Figure 20).

```
// ...
int process(int a, int b);
int main() {
    // ...
    c = process(a, b);          // processing
    // ...
}
int process(int a, int b) {
    // USER CODE BEGINS

    // USER CODE ENDS
}
```

Figure 20: Pre-written code template with subroutines for the example task

Since subroutines are functionally well separable units, with proper task description it is possible to solve the problem by implementing *only one* function:

```
int process(int a, int b) {
    // USER CODE BEGINS

    // USER CODE ENDS
}
```

Figure 21: Pre-written code showing only the main processing function

This pattern has many advantages methodologically. On the one hand, it shows very well that the solution of the problem is sharply separated from input and output management. Functions, as interfaces, communicate with the outside world through their parameters and return values, not being interested in where the data comes from and where it goes. They practice modularity, separation of responsibilities, and they can be tested well as the calling environment determines the input data and evaluates the outgoing. It is up to the programming language how this kind of interface should be implemented, what language elements are allowed. The most flexible solution can be achieved by utilizing the modular options of a particular programming language (modules, classes, functions). In C++, for example, the user code can be implemented in a header file that is loaded and used by the runtime or testing environment:

```cpp
// task.h
#include <vector>            // arbitrary includes
typedef vector<int> Numbers; // arbitrary types and constants
int process(int a, int b) {
    // USER CODE
}

// program.cpp
#include "task.h"
int main() {
    int a, b, c;             // declaration
    // ...
    c = process(a, b);       // processing
    // ...
}
```

Figure 22: C++ user solution in a separate module for the example task

5.3.3. Input-output data structure

Investigating the input-output possibilities in algorithms (Section 4), we saw the option of input and output as a complex data structure. For prepared inputs, there may be a variant which works with such data structures. It can work with and without subroutines, of course, but the following examples utilize subroutines. For C++, such processing may look like this:

```cpp
typedef int Integer;
struct Input { Integer a, b; };
struct Output { Integer c; };
Output process(Input input) {
    int a = input.a;         // "reading input"
    int b = input.b;
    int c = a + b;           // "processing"
    Output output;           // "writing output"
    output.c = c;
    return output;
}
```

Figure 23: C++ implementation of an input-output data structure for the example task

With modern language elements, "reading" and "writing" can be simplified as follows:

```cpp
Output process(Input input) {
    auto [a, b] = input;     // "reading input"
    int c = a + b;           // "processing"
    return { c };            // "writing output"
}
```

Figure 24: Reading and writing using complex data structures in C++ with modern language elements

For example, in TypeScript, it looks like this:

```
type Input = {a: number, b: number};
type Output = {c: number};
export function process({a, b}: Input): Output {  // "reading input"
    const c: number = a + b;                      // "processing"
    return { c };                                 // "writing output"
}
```

Figure 25: Reading and writing using complex data structures in TypeScript with modern language elements

5.4. Structured input

In the case of console programs, the input is practically the sequence of the requested data. Its structure is therefore determined by the logic of the requesting program. If this data is saved into a file, then it will contain a set of numbers and texts, which can be understood in the light of the requesting logic, but without it, it remains just a meaningless unstructured file. In addition, reading the input is done in a reverse order: a sequence of data is given, and this must be read into the appropriate data structures. One of the advantages of reading unstructured input is to leave the freedom of data representation. The disadvantage is the complicated logic for reading, and the lack of transparency when it comes to edit the input file.

Using structured input can help a lot on those disadvantages. This practically means using a human-readable format for describing the input structure. Among today's popular text-based data-serialization formats it is worth mentioning JSON and YAML.[4] Table 1 shows a comparison between these two format in the case of our example and for a more complex data structure.[5]

YAML	JSON

`a: 3  # first number`
`b: 5  # second number`
	`{`
	`  "a": 3,`
	`  "b": 5`
	`}`

`teachers:`
`  -`
`  name: Zsakó László`
`  code: zslzsl`
`  -`
`  name: Szlávi Péter`
`  code: szpszp`
	`{ "teachers": [`
	`  { "name": "Zsakó László",`
	`  "code": "zslzsl" },`
	`  { "name": "Szlávi Péter",`
	`  "code": "szpszp" }`
	`  ]`
	`}`

Table 1: Comparing YAML and JSON format for the same examples

How to read structured data depends on the programming language. For these popular formats, there is usually a library that makes their management easy. For C++, there are external libraries for both formats[6,7], and for TypeScript, JSON is a legal citizen of the language, because it is a serialized form of JavaScript (and thus TypeScript) language structures, and YAML→JSON

[4] The XML format is used rather for machine processing than for human editing. Other formats such as HOCON (https://github.com/lightbend/config/blob/master/HOCON.md) is exciting, but their support and familiarity is low.

[5] It is worth noting that from version 1.2 the JSON description is part of the YAML language.

[6] JSON for C++, https://github.com/nlohmann/json (utoljára megtekintve: 2017.11.11.)

[7] YAML for C++, https://github.com/jbeder/yaml-cpp (utoljára megtekintve: 2017.11.11.)

converter can be found as well.[8] Generally speaking, it is good if the structured format can be easily mapped to the data structures of the language.

Appendix 2 shows how to read JSON data in C++. The applied JSON library automatically recognizes simple types and vector types, but for complex structures, the developer (teacher or student) must write the conversion code. For this, the knowledge of the library is required. The example in the appendix contains an intentionally more complex input reading.

There is no need for extra libraries for TypeScript:

```
const input: Input = JSON.parse(textarea.value);
```

Figure 26: Reading JSON input in TypeScript

The structured input, as its name suggests, already contains the decisions needed to represent the data. If the input data is given in this form, then the student does not practice the data representation. Structured inputs can also be used in automatic evaluation environments with proper preparation [3,4].

5.5. Input and output as a separate task

At the beginning of this article, we mentioned that during coding, the student actually meets three tasks: reading the data, calculating the result, and writing it out. So far, we have been focusing on how we can concentrate more on problem solving instead of input management. However, reading the input can also be a separate task: new language elements, concepts and techniques can be taught with it, and in more complex cases special programming theorems can be used during solution. For example, reading an array is a mapping programming theorem, while in precondition checking, any programming theorem (most often a decision) can occur. It can be given as task to student to read a given input file into a specific data structure, or to find out the data structure himself or herself. Thus, the input reading itself becomes processing, and the result of the reading must be printed to screen. Otherwise, it may have a positive benefit to force the students check the read data.

There has been little talk about the output part, because its complexity is usually much less than reading's. Less language items are required also. Thus, these kinds of tasks can come up relatively early, even at the very beginning. In these tasks, a data structure s given baked-in and this should be printed to the screen. Initially, these are simple types and can be written with a single command, later a selection is needed for writing logical values out, and an iteration for arrays. The applied programming theorems, if any, is almost always a mapping.

6. Conclusion

Nowadays, in programming education, we often find it natural that reading and writing is an integral part of problem solution, together with their complexities. Although this is a legitimate expectation looking at the end result, this article tried to show that this is not always necessarily the case. The three tasks can be separated from each other and gradually introduced. As we gradually progress towards the more difficult examples, it is also advisable to familiarize students with the more complex cases of input reading, when the students are prepared for its complexity. The three major subtasks of the solution (reading-processing-writing) do not have to go hand in hand in each task, their separation also allows us to deal with more tasks from the curriculum. For most tasks, omit the input or make it very simple, and only sometimes go into details, for

[8] YAML for JavaScript, https://github.com/nodeca/js-yaml (utoljára megtekintve: 2017.11.11.)

input-specific tasks or home assignments. This article showed a number of ways how to make input management easier. Each method also allows the input handling to be progressively introduced, for example, at first with baked-in data, then using prepared code templates, and finally with input-only tasks. For most tasks, however, it is advisable to completely ignore the input-output part. Most methods can work in traditional development environments, but learning can also be supported in special environments that provide automatic evaluation or editing capabilities.

References

1. Linda Mannila, Mia Peltomäki & Tapio Salakoski: *What about a simple language? Analyzing the difficulties in learning to program*, Computer Science Education, Vol. 16, No. 3, September 2006, pp. 211 – 227. DOI: 10.1080/08993400600912384
2. Eric Roberts: *An Overview of MiniJava*, Conference Paper in ACM SIGCSE Bulletin, March 2001. DOI: 10.1145/366413.364525
3. Horváth Győző, Menyhárt László: *Webböngészőben futó programozási környezet megvalósíthatósági vizsgálata*, INFODIDACT 2016, Zamárdi, 2016.
4. Győző Horváth: *A web-based programming environment for introductory programming courses in higher education*, Annales Mathematicae et Informaticae, 48 (2018) pp. 23–32

Appendix

1. An example for a more complex input reading in C++

```
#include <iostream>
using namespace std;
int main(){
  // declaration
  int a, b;
  int c;

  // reading input
  bool good;
  string sv;
  do {
    clog << "First number: ";
    cin >> a;
    good = cin.good() && (a>0);
    if (!good) {
      clog << "The number should be positive!" << endl;
      cin.clear();
      getline(cin, sv);
    }
  } while(!good);

  do {
    clog << "Second number: ";
    cin >> b;
    good = cin.good() && (b>=0);
    if (!good) {
      clog << "The number should be non-negative!" << endl;
      cin.clear();
      getline(cin, sv);
    }
  } while(!good);

  // processing
  c = a + b;

  // writing output
  cout << c << endl;

  return 0;
}
```

2. A complex example for reading JSON in C++

```
#include <iostream>
#include <vector>
#include "json.hpp"
using json = nlohmann::json;
using namespace std;

typedef vector< vector<int> > matrix;
struct Point { int x, y; };
```

```
struct Input {
    int a, b;
    vector<int> numbers;
    vector<Pont> arrayOfPoints;
    matrix m;
};
void from_json(const json& j, Pont& p) {
    p.x = j.at("x").get<int>();
    p.y = j.at("y").get<int>();
}
void from_json(const json& j, Input &input) {
    input.a            = j.at("a");
    input.b            = j.at("b");
    input.numbers      = j.at("numbers")      .get< vector<int> >();
    input.arrayOfPoints= j.at("arrayOfPoints").get< vector<Pont> >();
    input.m            = j.at("m")            .get<matrix>();
}
void process(Input input) {
    auto [a, b, numbers, arrayOfPoints, m] = input;
}
int main() {
    json j;                 // reading input
    cin >> j;
    Input input = j;
    process(input);         // processing
    // ...                  // writing output
}
```

Authors

HORVÁTH Győző

Eötvös Loránd University, Faculty of Informatics, Department of Media and Educational Informatics, Budapest, Hungary, e-mail: gyozo.horvath@inf.elte.hu

About this document

Published in:

CENTRAL-EUROPEAN JOURNAL OF NEW TECHNOLOGIES IN RESEARCH, EDUCATION AND PRACTICE

Volume 1, Number 1. 2019.

ISSN: 2676-9425 (online)

DOI:

10.36427/CEJNTREP.1.1.382

License