Programovanie virtuálneho objektu prístupné pre nevidiacich žiakov sekundárneho vzdelávania

Rebeka Sojka

Diplomová práca

- Školiteľka: doc. RNDr. Ľudmila Jašková, PhD.
- Cieľ práce: programovacie prostredie pre nevidiacich; pohyb virtuálneho robota po ozvučenej štvorcovej sieti
- Cieľová skupina: 2. stupeň ZŠ

Prístupné programovacie prostredia

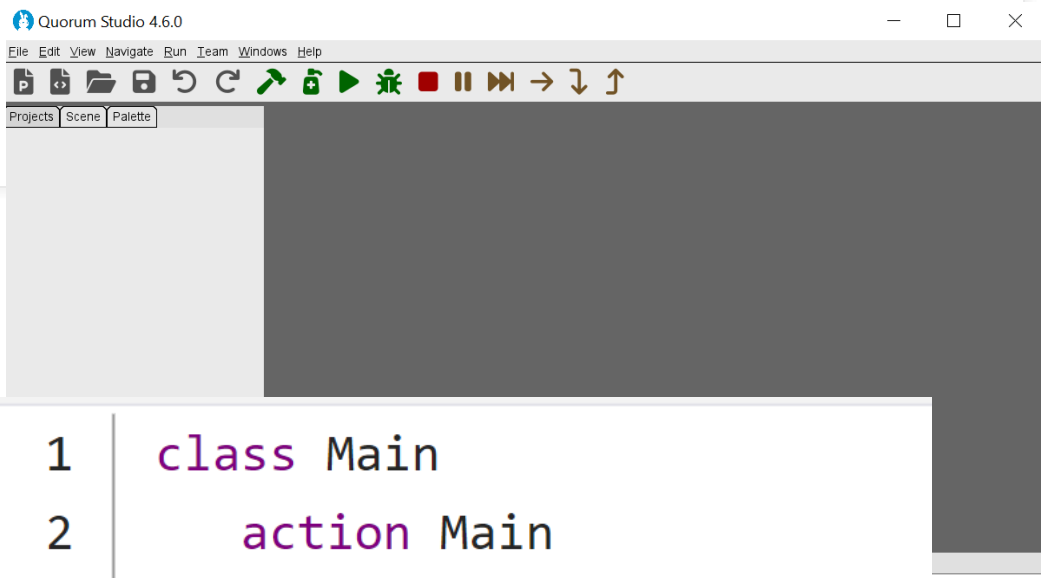
Quorum

Code Jumper

Coshi 2019

Quorum (jazyk aj IDE)

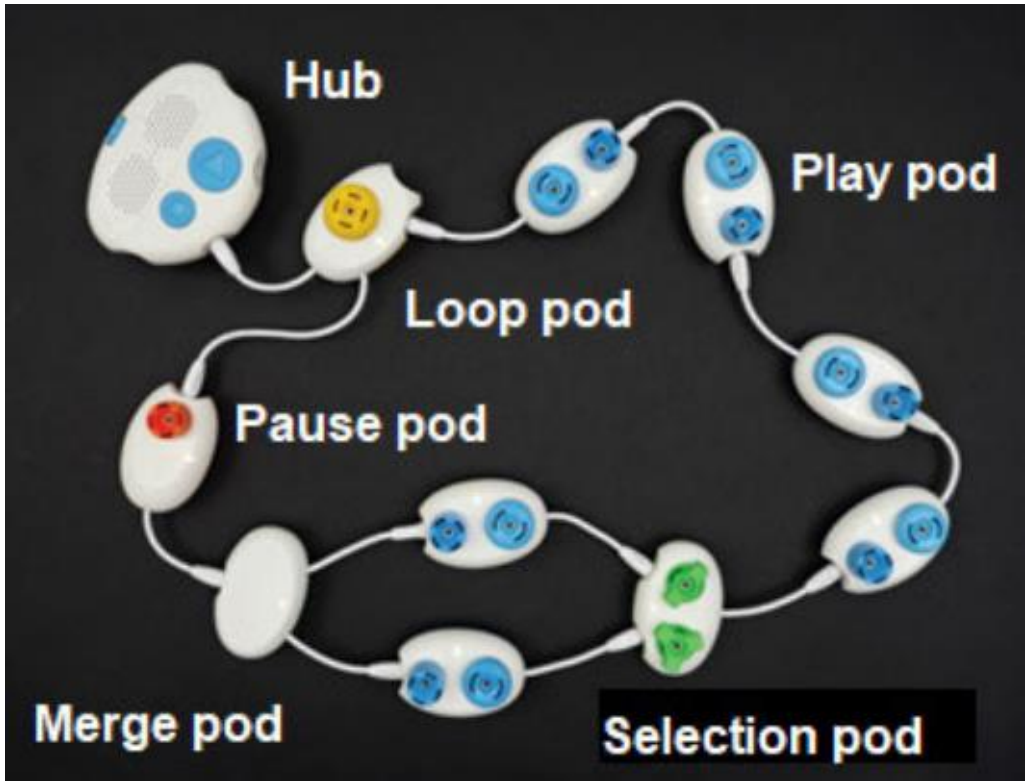
- Online tutoriál
- Stiahnuteľné prostredie
- Podpora hovoreného výstupu
- Podpora hlasových príkazov
- Knižnice

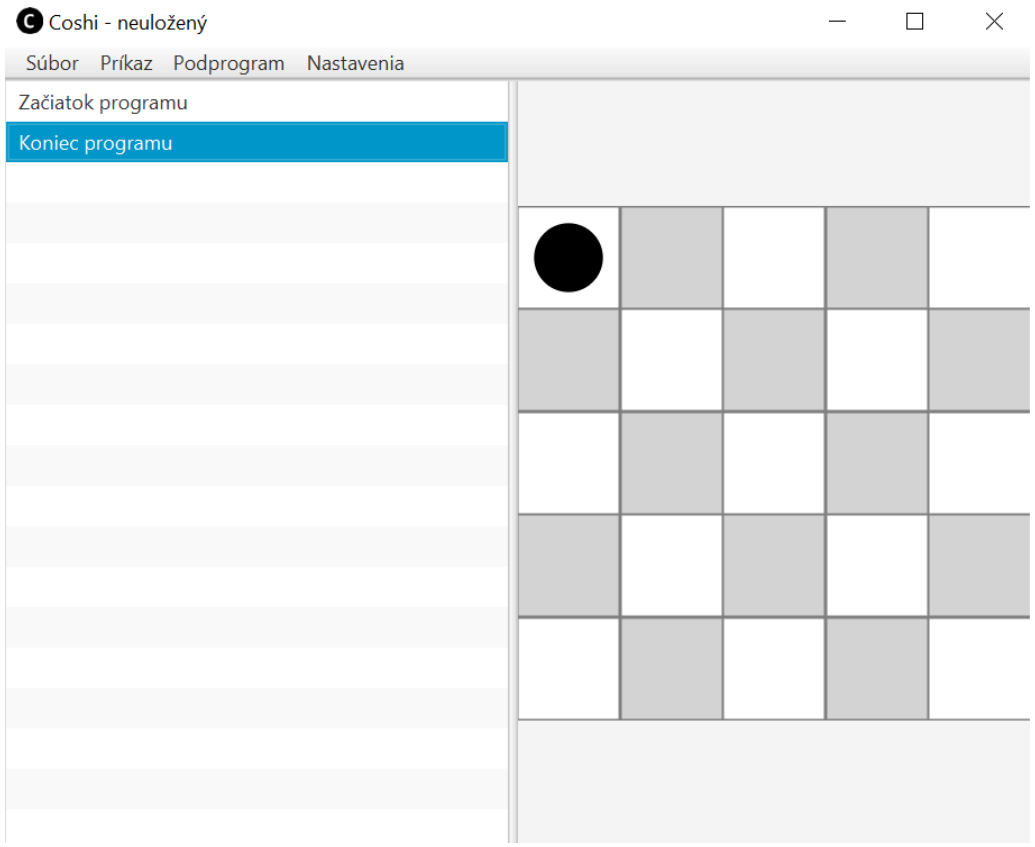



```
1  class Main
2      action Main
3          integer i = 15
4          //integer j = 25
5          //integer k = 35
6          output i
7      end
8  end
```

Code Jumper

- Set fyzických zariadení
- Prístupné blokové programovanie
- Rozvíja prácu vo dvojiciach
- Motivácia využitia cyklov





Coshi, 2019

- Skladanie programu z hotových príkazov
- Bez terminálu
- Syntax jazyka
- Navigácia v kóde
- Klávesové skratky
- Pre nižšie ročníky

Postup práce 1. ročník

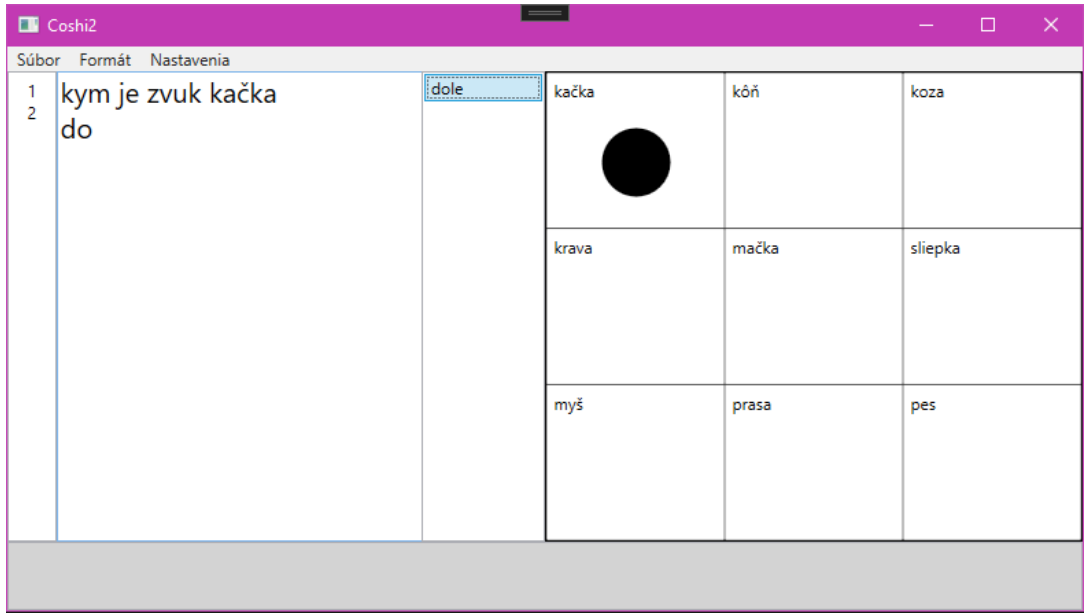
1. Výber technológií C# a .NET
2. Malá testovacia aplikácia
3. Syntax jazyka
4. Literatúra
5. Interpreter pre základné príkazy a for cykly
6. Terminál

Postup práce 2. ročník

7. Kompilátor s virtuálnym strojom, while cykly
8. Práca s premennými, vetvenia
9. Zvukové balíčky, nastavenia
10. Podprogramy
11. Funkcie pre zjednodušenie prístupu, pohyb robota
12. Práca so súbormi

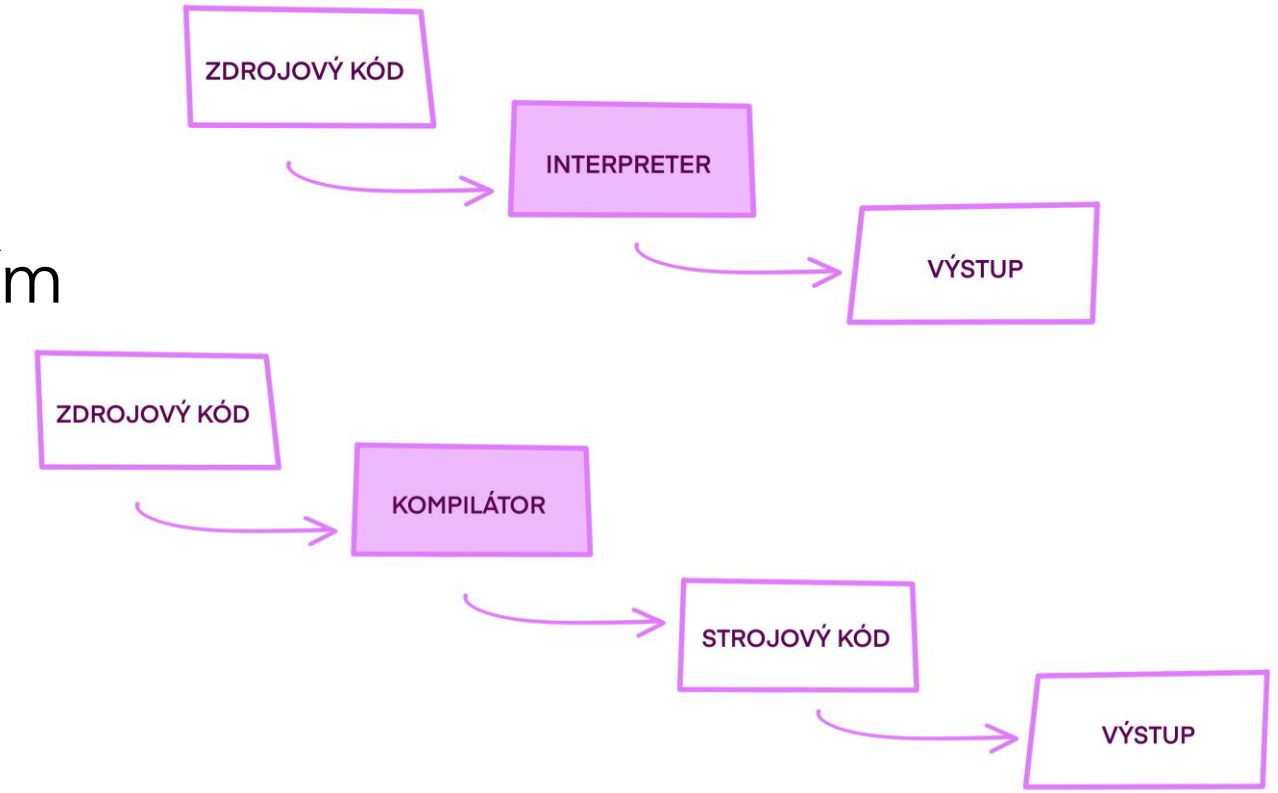
Postup práce 2. ročník

- Optimalizácia riešení
- Prvotné testovanie
- Výskum v spolupráci s testovacou skupinou
- Agilná fáza vývoja softvéru



Implementácia – Kompilátor

- Virtuálna mašina
- Medzikód, bytecode
- Hlásenie chýb pred spustením
- Podprogramy



Implementácia - Virtuálna mašina

- Pamäť – pole celých čísel mem
- Stromová štruktúra
- Zásobník
- Procesor
- Inštrukcie a ich kódy

```
1 reference
public static void execute()
{
    if (mem[pc] == INSTRUCTION_PUSH)
    {
        pc = pc + 1;
        top = top - 1;
        mem[top] = mem[pc];
        pc = pc + 1;
    }
    else if (mem[pc] == INSTRUCTION_MINUS)
    {
        pc = pc + 1;
        mem[top] = -mem[top];
    }
    else if (mem[pc] == INSTRUCTION_ADD)
    {
        pc = pc + 1;
        mem[top + 1] = mem[top + 1] + mem[top];
        top = top + 1;
    }
    else if (mem[pc] == INSTRUCTION_SUB)
    {
        pc = pc + 1;
        mem[top + 1] = mem[top + 1] - mem[top];
        top = top + 1;
    }
    else if (mem[pc] == INSTRUCTION_GET)
    {
```

Opakuj 2 krát vpravo dole koniec

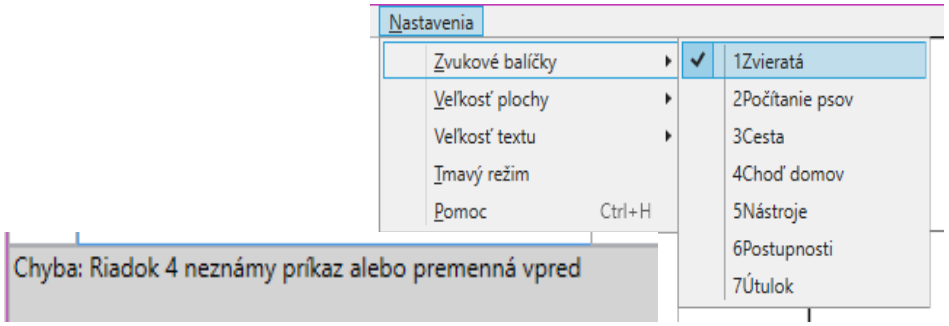
mem[0] = 9	inštrukcia JUMP
mem[1] = 2	na adresu 2
mem[2] = 13	inštrukcia PUSH ...
mem[3] = 2	číslo 2 na koniec zásobníka
mem[4] = 4	inštrukcia vpravo
mem[5] = 2	inštrukcia dole
mem[6] = 7	inštrukcia LOOP
mem[7] = 4	adresa začiatku tela cyklu

```
INSTRUCTION_JUMP = 9;
INSTRUCTION_DOWN = 2;
INSTRUCTION_PUSH = 13;
INSTRUCTION_RIGHT = 4;
INSTRUCTION_LOOP = 7;
```

ZÁSOBNÍK

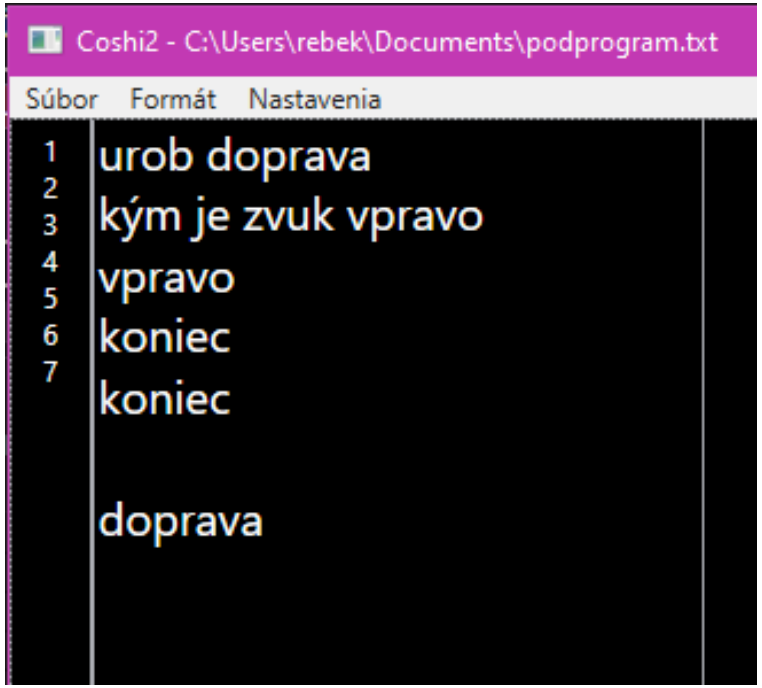
Implementácia – Prístupnosť

- Prieskum hracej plochy
- Skákanie po blokoch kódu
- Predikcia
- Nastavenia
- Chybové hlášky



Implementácia – Obmedzenia

- Syntax
- Definícia pred volaním
- Predikcia



Implementácia – Krátka ukážka

Literatúra

Mini-languages: A way to learn programming principles

„General-purpose languages provide little leverage for understanding their basic actions and control structures. The languages are not visual and their basic functions are carried out behind an opaque barrier. Professional programming environments do not usually provide any visualization capabilities. With the process of program execution hidden, the student develops an input–output orientated understanding."

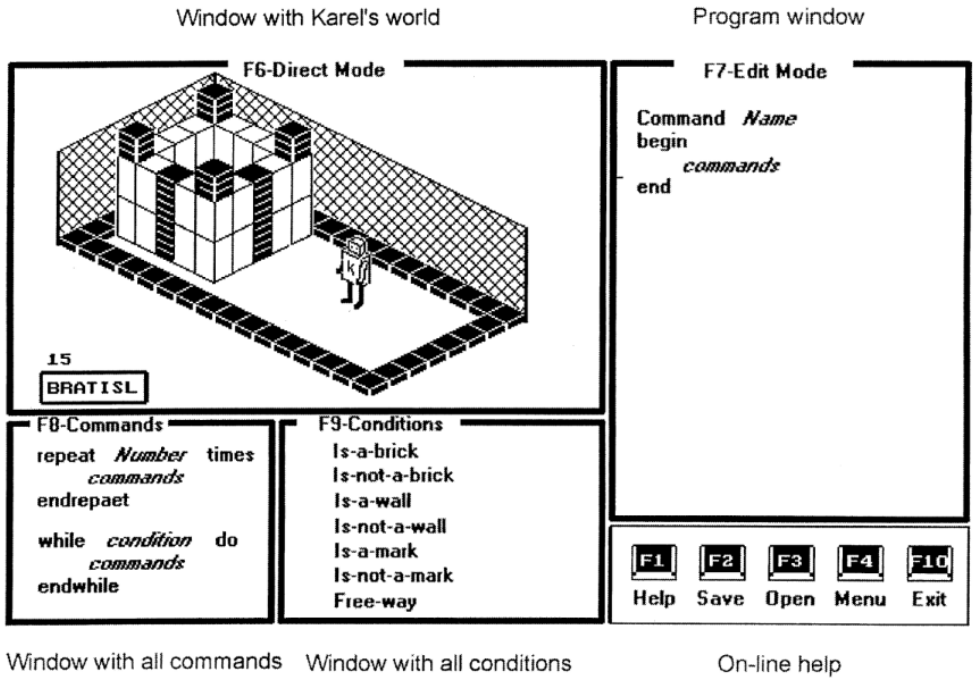


Figure 2. Karel-3D

Autori: PETER BRUSILOVSKY, EDUARDO CALABRESE, JOZEF HVORECKY, ANATOLY KOUCHNIRENKO, PHILIP MILLER

Making Programming Accessible to Learners with Visual Impairments: A Literature Review

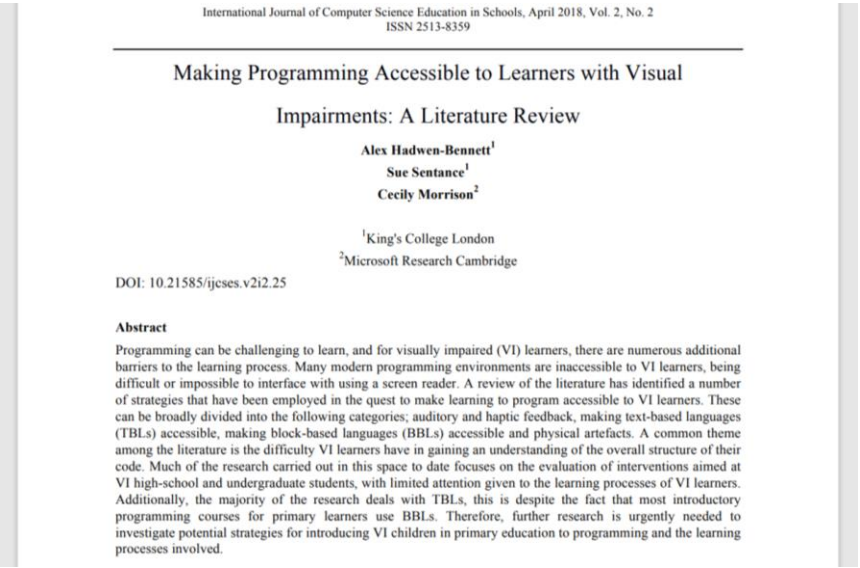
„When teaching with text-based programming languages, the choice of language is important. Either choose a language that is specially designed for VI learners, or a general-purpose language with simple syntax and limited use of non-alphanumeric characters, for example Ruby. “

Authors:

Alex Hadwen-Bennett
King's College London

Sue Sentance
King's College London

Cecily Morrison



International Journal of Computer Science Education in Schools, April 2018, Vol. 2, No. 2
ISSN 2513-8359

Making Programming Accessible to Learners with Visual Impairments: A Literature Review

Alex Hadwen-Bennett[1]
Sue Sentance[1]
Cecily Morrison[2]

[1]King's College London
[2]Microsoft Research Cambridge

DOI: 10.21585/ijcses.v2i2.25

Abstract

Programming can be challenging to learn, and for visually impaired (VI) learners, there are numerous additional barriers to the learning process. Many modern programming environments are inaccessible to VI learners, being difficult or impossible to interface with using a screen reader. A review of the literature has identified a number of strategies that have been employed in the quest to make learning to program accessible to VI learners. These can be broadly divided into the following categories; auditory and haptic feedback, making text-based languages (TBLs) accessible, making block-based languages (BBLs) accessible and physical artefacts. A common theme among the literature is the difficulty VI learners have in gaining an understanding of the overall structure of their code. Much of the research carried out in this space to date focuses on the evaluation of interventions aimed at VI high-school and undergraduate students, with limited attention given to the learning processes of VI learners. Additionally, the majority of the research deals with TBLs, this is despite the fact that most introductory programming courses for primary learners use BBLs. Therefore, further research is urgently needed to investigate potential strategies for introducing VI children in primary education to programming and the learning processes involved.

THREE PROGRAMMING ENVIRONMENTS FRIENDLY TO BLIND STUDENTS IN LOWER SECONDARY EDUCATION

„Generally, we can say that the Coshi environment has been proven as appropriate for blind students. It has a good sound response and the creation of a program using commands from a menu or using the keyboard shortcuts was suitable for students. The commands were understandable, but it is important to implement a new command – conditional statement without the ELSE branch.“

Autorka: ĽUDMILA JAŠKOVÁ

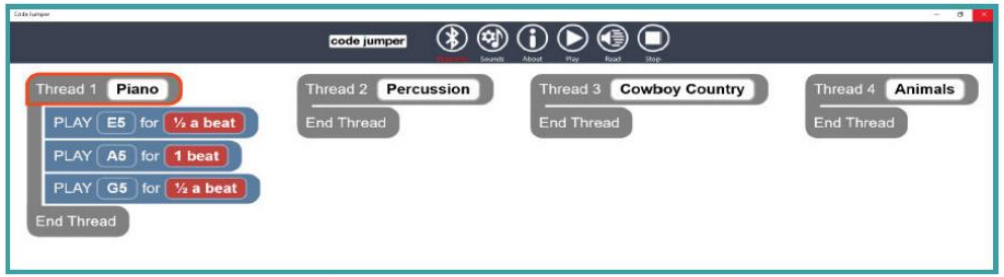


Figure 2. A software running on the computer with a text (block-based) version of program[4].

While the Code Jumper was created, our team developed the Alan environment in collaboration with our student of applied informatics [3].

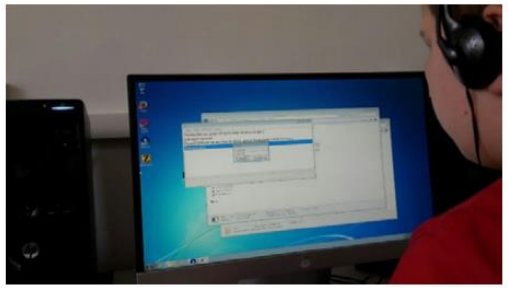


Figure 3. Blind student programming in Alan.

The Alan environment allows a user to program audio stories or simple melodies. The environment uses a simple programming language with two basic commands: **Play** and **Say**.

Ďakujem za pozornosť

Rebeka Sojka