

Mestna občina Celje  
Komisija Mladi za Celje

# **Python, java in C# na Raspberry Pi**

RAZISKOVALNA NALOGA

AVTORJI

Urban Vidmar

Jaša Pelc

Nik Prodanović

MENTOR

mag. Boštjan Resinovič, univ. dipl. inž

Šmarje pri Jelšah, marec 2015

Šolski Center Celje,  
Srednja šola za kemijo, elektrotehniko in računalništvo

# **Python, java in C# na Raspberry Pi**

RAZISKOVALNA NALOGA

AVTORJI

Urban Vidmar

Jaša Pelc

Nik Prodanović

MENTOR

mag. Boštjan Resinovič, univ. dipl. inž

Mestna občina Celje, Mladi za Celje

Šmarje pri Jelšah, marec 2015

# Kazalo

Povzetek .....	1
Abstract .....	1
1 Uvod .....	3
1.1 Opredelitev problema .....	3
1.3 Hipoteze .....	4
1.4 Izbira raziskovalnih metod .....	4
2 Nekaj o Raspberry Pi-ju .....	5
3 Načrtovanje .....	6
3.1 Raspberry Pi .....	6
3.2 Programi .....	6
4 Izvedba .....	7
4.1 NOOBS .....	7
4.2 Java .....	8
5.3 C# .....	13
5.4 Python .....	22
6 Testiranje .....	25
7 Analiza rezultatov .....	28
Zaključek .....	31
Zahvala .....	32
Viri .....	33

## Kazalo slik, tabel in grafov

Slika 1: Raspberry Pi B+ .....	5
Slika 2: NOOBS .....	6
Slika 3: Izbirno okno OS .....	7
Slika 4: Inštalacija OS .....	8
Slika 5: Preverjanje nameščene verzije java .....	8
Slika 6: Prenos najnovejše java .....	8
Slika 7: Inštalacije java .....	9
Slika 8: Program z izpisom števil .....	10
Slika 9: Slika programa z barvami .....	11
Slika 10: Primer sintakse okenskega programa v javi .....	12
Slika 11: izgled okenskega programa v javi .....	12
Slika 12: Zagon LXTerminal .....	13
Slika 13: Okno LXTerminal .....	13
Slika 14: Nameščanje emacs .....	14
Slika 15: C# v konzoli .....	14
Slika 16: Osnovni program v C# .....	15
Slika 17: MonoDevelop .....	16
Slika 18: Izvajanje programa v LXTerminal .....	16
Slika 19: Koda v LeafPadu .....	17
Slika 20: Program z generiranjem naključnih števil .....	18
Slika 21: Razvrščanje barv po abecedi .....	19
Slika 22: Grafični program - kalkulator – 1 C# .....	20
Slika 23: Grafični program - kalkulator – 2 C# .....	20
Slika 24: Sintaksa grafičnega programa C# .....	21
Slika 25: Ikona za zagon IDLE 3 .....	22
Slika 26: Testni program python .....	22
Slika 27: Program za izpis in razvrščanje števil .....	23
Slika 28: Program z razvrščanjem števil .....	23
Slika 29: Izgled grafičnega programa .....	24
Slika 30: Sintaksa grafičnega programa python .....	24
Slika 31: Hello World v python in java .....	25
Tabela 1: Osnovne karakteristike .....	5
Tabela 2: Čas izvajanja java .....	26
Tabela 3: Čas izvajanja C# .....	26
Tabela 4: Čas izvajanja Phytion .....	26
Tabela 5: Čas izvajanja na Raspberry Pi .....	26
Tabela 6: Čas izvajanja na PCju .....	26
Graf 1: Grafični prikaz časa izvajanja za program 1 .....	27
Graf 2: Grafični prikaz časa izvajanja za program 2 .....	27

## **Povzetek**

Za temo naše raziskovalne naloge smo se odločili predvsem zato, ker menimo, da majhni računalniki postajajo vse bolj popularni, med navadnimi uporabniki kot med programerji. Zato smo se odločili preveriti ali se lahko Raspberry Pi primerja z osebnim računalnikom, v izvajanju programov ter samem programiranju. Najprej smo pripravili Raspberry Pi za uporabo. Nato smo se seznanili z našimi programskimi jeziki c#, python ter java. Z našim znanjem pridobljenim iz različnih virov o naših programskih jezikih ter Raspberry Pi-ju je vsak izmed nas napisal tri različne programe. Pred izvajanjem programov smo si postavili hipoteze katere bomo po pridobljenih rezultatih ovrgli ali pa potrdili. Programe smo preizkusili na osebni računalniku ter RPi-ju. Na podlagi pridobljenih rezultatov ter našega predznanja smo začeli potrjevati ter ovračati hipoteze, ki smo si jih bili prej zastavili. Nekaj smo jih potrdili nekaj pa tudi ovrgli. Nad nekaterimi ugotovitvami smo bili zelo presenečeni.

## **Abstract**

The main purpose for choosing the theme is our opinion on small computers becoming more popular among mundane users and programmers alike. We decided to check if Raspberry Pi can stand up to personal computer, in executing programs and also in programing programs. First we prepared Raspberry Pi for our use. Then we decided to get to know our programing languages better C#, python and java. With our knowledge gained from different sources about our programing languages and Raspberry Pi, each one of us wrote three different programs. Before executing programs we set hypotheses which we will either confirm or dismiss after getting results. We tested programs on personal computer and RPi. On the basis of our findings we dismissed and/or confirmed our hypotheses set before. Some of the results were quite impressing.

## **Ključne besede**

RPi = Raspberry Pi – mini računalnik v velikosti kreditne kartice

Java – najbolj razširjeni programski jezik (routerji, telefoni ...)

Python – programski jezik, ki je privzet na Raspberry Pi

C# – programski jezik

OS – operacijski sistem

NOOBS – grafični vmesnik za izbiro operacijskega sistema

PC (ang. Personal computer) – osebni računalnik

# 1 Uvod

V zadnjih letih se je na trgu pojavilo kar nekaj miniaturnih računalnikov, ki na začetku niso pritegnili veliko pozornosti. V zadnjem času pa so se ti miniaturni računalniki razširili predvsem za učne namene. Morda ste že slišali za katerega izmed njih - Banana Pi, Raspberry Pi ali Odroid? Lahko bi jih našteali še kar nekaj, vendar smo omenili samo najbolj znane.

Najbolj razširjen med njimi in znan je prav gotovo Raspberry Pi, ki se je uveljavil na trg predvsem zaradi svojega uporabniškega vmesnika, ki je začetnikom enostaven, hkrati pa tudi poučen. Predvsem smo zaradi naše stroke imeli v mislih osnove programiranja, ki se jih učimo že kar nekaj let. Prav zaradi tega smo se odločili, da bomo v tej raziskovalni nalogi uporabili Raspberry Pi in ga združili z našim predznanjem o programskih jezikih.

Pri testiranju smo se odločili, da bomo za to uporabili različne programske jezike. Prvi izmed njih je python, ki je že vnaprej naložen na Raspberry Pi. Odločili smo se tudi za java, ki je zelo razširjena na vseh področjih računalništva (routerji, mobilni telefoni ...) in ker je že na Računalniku v osnovi naložena. Kot tretji in zadnji programski jezik, ki smo ga uporabili, je C#, ker je objektno orientiran programski jezik. Kot primerjavo Raspberry Pi-u smo uporabili nam vsakdanji osebni računalnik, ker je Raspberry Pi na nek način njegova pomanjšana in cenejša verzija.

## 1.1 Opredelitev problema

V naši raziskovalni nalogi bomo poskušali ugotoviti, ali se splača naložiti java in C# na Raspberrya ali je bolje delati kar v pythonu, ki je že nameščen. Zanima nas tudi kateri izmed, teh programskih jezikov se najboljše obnese in seveda kateri deluje hitreje. Hkrati pa jih bomo testirali tudi na PC-ju, da ugotovimo, kako se RPi obnese v primerjavi z osebnim računalnikom.

### **1.3 Hipoteze**

Pred samim začetkom našega raziskovanja smo si zadali hipoteze, ki jih bomo poskušali skozi naše raziskovanje potrditi oziroma ovreči:

- Najprimernejši jezik za Raspberry Pi je python.
- Raspberry Pi lahko v prihodnosti konkurira osebnim računalnikom.
- C# je primeren za delo na Raspberry Pi.
- Java je učinkovita pri delu na Raspberry Pi.
- Programi na Raspberry Pi se hitreje izvajajo kot na osebnem računalniku.
- Raspberry Pi je primeren pripomoček za izvajanje poučevanja programiranja.
- Na Raspberry Pi se splača izdelovati grafične aplikacije.

### **1.4 Izbira raziskovalnih metod**

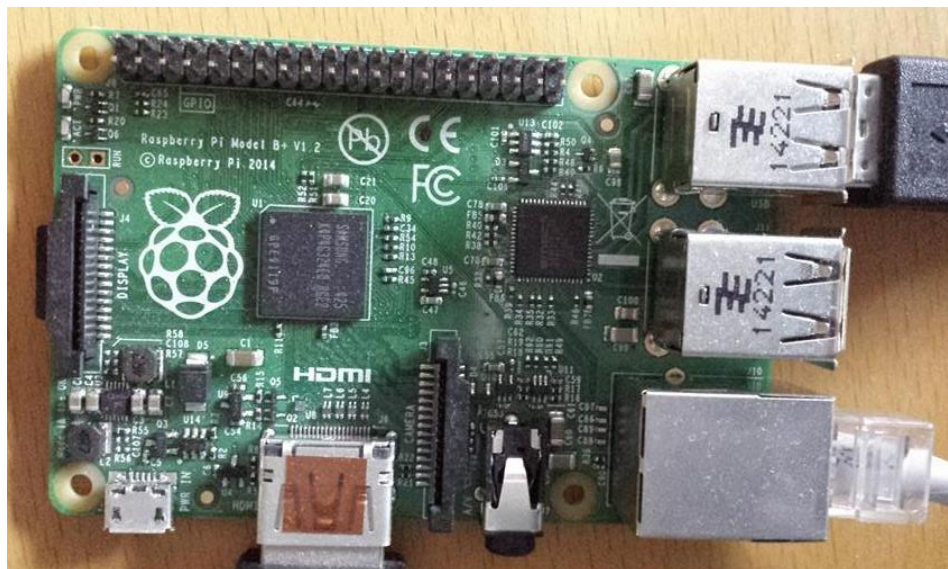
Za izhodišče naše raziskovalne naloge bomo uporabili miniaturni računalnik Raspberry Pi verzije B+. Za naše delo se bomo najprej lotili učenja programskih jezikov java, python in C#. Na Raspberry Pi bomo najprej naložili operacijski sistem Raspbian, nato se bomo lotili izdelovanja treh programov, katerih namen je ugotoviti, kje se izvajajo hitreje na osebnem računalniku ali Raspberry Pi-u. Za primerjavo bomo uporabili navadni osebni računalnik, ker je lahko dostopen in ga uporabljamo vsi. Na obeh napravah bomo testirali različne programe in izračunali čas njihovega izvajanja. Odločili smo se, da bomo v vsakem izmed jezikov izdelali dve konzolni aplikaciji in eno grafično. Pri analizi bomo upoštevali tudi čas in zahtevnost razvoja posameznih programov.



## 2 Nekaj o Raspberry Pi-ju

Marsikdo se vpraša: »Kaj je to Raspberry Pi?«

Odgovor na to je zelo preprost. To je računalnik. Če razložimo malo podrobneje je to miniaturni računalnik v velikosti bančne kartice. Ima USB priključke za priključitev miške, tipkovnice, USB ključkov ipd. Na isti plošči najdemo še HDMI priključek za zaslon, procesor, grafično kartico, RAM pomnilnik, priključitev za napajanje, avdio priključek, 40-pinski priključek ter priključek za kamero. Celoten operacijski sistem ter spomin pa se nahajajo na spominski kartici.



Slika 1: Raspberry Pi B+

Najbolj priljubljen OS za RPi je Raspbian. Vsi OS pa temeljijo na Linuxovem jedru kernel. Za OS je potrebno vsaj 4 GB prostora. Kot primarno orodje na RPi-ju je programski jezik python.

Prvi model tega računalnika je prišel na trg v februarju 2012. Na trgu so trenutno na voljo 3 modeli Raspberry Pi-ja (A+, B, B+). Model A+ stane 20 \$, model B in B+ pa 35 \$. Mi smo za preizkušanje in testiranje uporabili model B+.

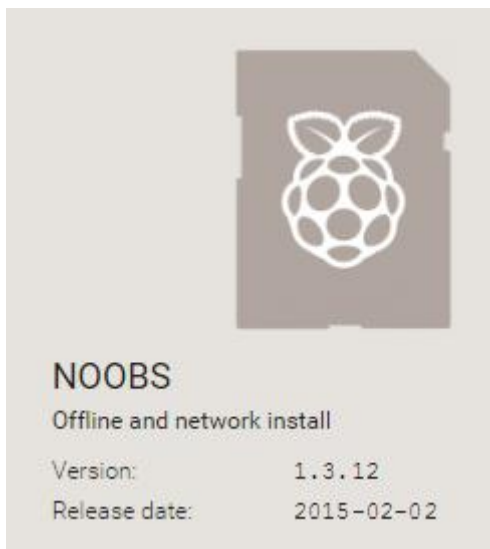
Tabela 1: Osnovne karakteristike

	A+	B	B+
cena	20\$	35\$	
CPU	700MHz ARM1176JZF-S (ARM 11 družina)		
GPU	Broadcom VideoCore IV, 250MHz OpenGL ES 2.0 (24 GFLOPS)		
SDRAM	256MB (deljen z GPU)	512MB (deljen z GPU)	
USB 2.0	1X	2X	4X
Spominska kartica	Micro SD	SD	Micro SD
moč	200mA (1 W)	700mA (3,5 W)	600mA (3 W)
Internet	/	10/100 Mbit/s	

## 3 Načrtovanje

### 3.1 Raspberry Pi

Za začetek smo morali na RPi naložiti ustrezen operacijski sistem. Odločili smo se za Raspbian, saj ima že naloženo java (od 2014-01-07) in python. Raspbian je na voljo za prenos na uradni strani Raspberry Pi. Na voljo je posamezno in v NOOBS (je kot privzeti OS). Je najbolj priljubljen in razširjen OS za RPi.



Slika 2: NOOBS

### 3.2 Programi

Kot smo omenili že v prejšnjih delih naše raziskovalne naloge, smo za naše raziskovanje in testiranje Raspberry Pia uporabili različne programe, napisane v programskih jezikih java, C# in python. Za primerjavo smo te programe izvedli tudi na namiznem računalniku.

#### PROGRAM S ŠTEVILI:

Ta program vsebuje polje  $n$  števil ki so naključno izbrana med 0 in 100. Ta števila se shranijo v polje števil velikosti  $n$ . Števila se nato izpišejo, tako da gremo s pomočjo zanke skozi celotno polje števil in jih izpišemo. Odločili smo se, da bomo imeli polje 100 000 števil ( $n = 100\,000$ ). Ta števila smo nato razvrstili od najmanjšega proti največjemu. Za pomoč pri testiranju smo uporabili v program vgrajeni števec, ki je računal čas izvedbe celotnega programa.

#### PROGRAM S STRINGI (BESEDAMI):

Program vsebuje polje stotih besed, ki jih kopira v novo polje in jih nato razvrsti po abecedi od a do ž. Mi smo za ta program uporabili polje z vnaprej vnesenimi imeni desetih različnih barv. Program je nato teh 10 barv izpisal 100 000-krat. Tako kot pri programu, ki dela s

števíli, smo tudi v ta program vnesli števec, ki je izračunal čas izvajanja celotnega programa, kar smo uporabili pri našem nadaljnjem testiranju.

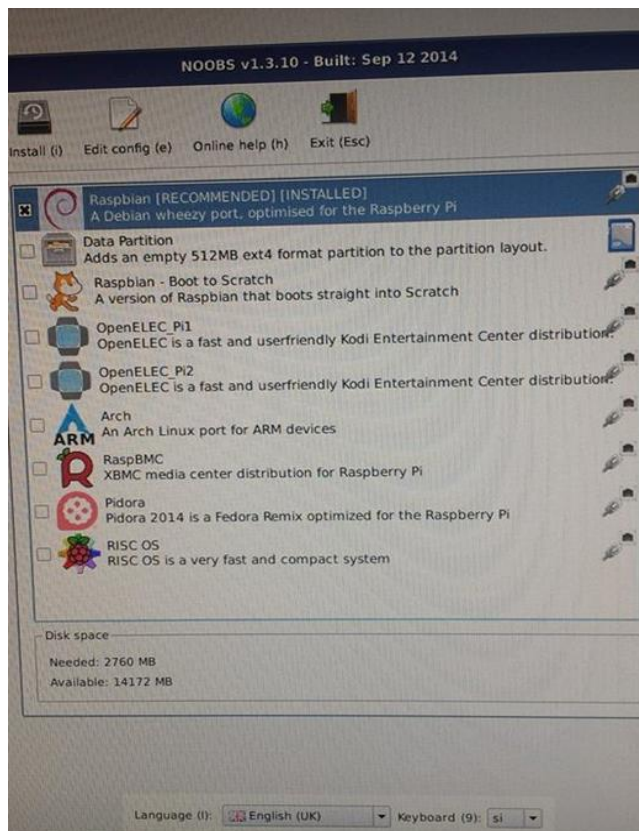
#### GRAFIČNI PROGRAM:

Izdelali smo tudi enostavne grafične oziroma okenske aplikacije. V tem primeru smo izdelali kalkulator, ki sešteje dve poljubni števíli. Tako kot pri prejšnjih dveh programih smo naredili programe v vseh treh jezikih, vendar smo uporabili nekoliko drugačen kriterij, katerega namen ni bil testiranje hitrosti, ampak ugotavljanje same zahtevnosti razvijanja takšnih aplikacij v posameznem jeziku. Pri tem smo upoštevali čas izdelave aplikacije, dolžino programa in samo zahtevnost uporabe razvojnega okolja ter sintakse programskega jezika.

## 4 Izvedba

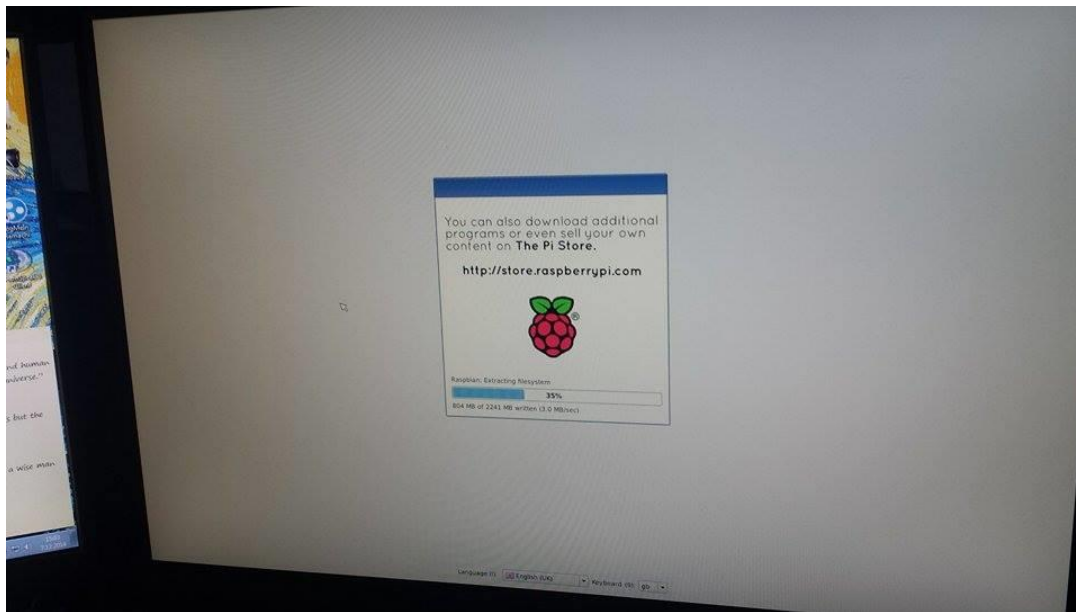
### 4.1 NOOBS

Na 16 GB veliko Micro SD kartico smo naložil NOOBS verzije 1.3.12 (priporočena velikost je 8 GB). NOOBS nam bo naložil Raspbian OS, lahko pa tudi izbiramo med drugimi operacijskimi sistemi, kot so: RISC OS, Piadora, RaspBMC ... Datoteko, ki jo prenesemo z interneta, razširimo in jo prenesemo na kartico (priporočljivo je formatirati kartico). Ko vse prenesemo, varno odstranimo kartico. Tako je kartica pripravljena. Nato si pripravimo Raspberry Pi (priključimo miško, tipkovnico ter ostale stvari) in vstavimo kartico, ki vsebuje NOOBS.



Slika 3: Izbirno okno OS

Ob prvem zagonu se pojavi okno za izbiro OS. Izbrali smo Raspbian OS, nato pa izbrali še, kateri jezik OS želimo ter kateri jezik tipkovnice bomo uporabili. Ko smo vse nastavili, smo kliknili naprej (ang. Next). Začne se nameščanje OS, zato počakamo, da se namestitev dokonča.



Slika 4: Inštalacija OS

Ob vsakem zagonu bo Pi najprej preveril sistem ter I/O, nato pa bo vprašal za uporabniško ime(pi) in geslo (Raspberry), ki je že nastavljeno in je na vsakem RPi-ju enako. Ko vneseš, lahko nadaljuješ in uporabljaš ukaze za linux. Če želimo, lahko zaženemo grafični način. To storimo tako, da vpišemo startx.

## 4.2 Java

V primeru, da smo naložili Raspbian, ki že vsebuje java, moramo samo preveriti, če imamo naloženo najnovejšo java. To lahko naredimo v Terminalu z ukazom **java-version**.

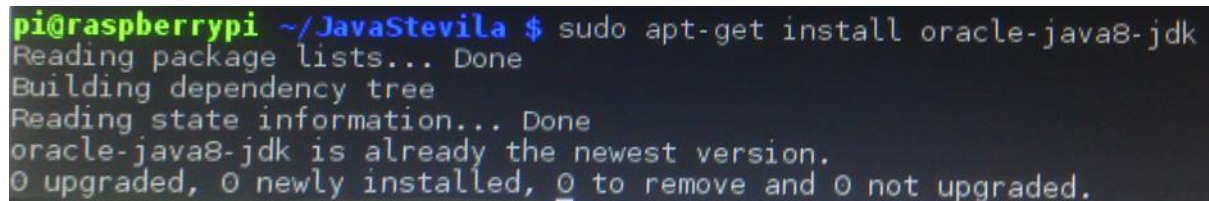
```
pi@raspberrypi ~/JavaStevila $ java -version
java version "1.8.0"
Java(TM) SE Runtime Environment (build 1.8.0-b132)
Java HotSpot(TM) Client VM (build 25.0-b70, mixed mode)
```

Slika 5: Preverjanje nameščene verzije java

```
pi@raspberrypi ~/JavaStevila $ sudo apt-get update
```

Slika 6: Prenos najnovejše java

Če pa smo naložili starejši Raspbian, moramo java naložiti ročno. To storimo preko Terminala. Obvezno moramo imeti vzpostavljeno internetno povezavo. Nato v Terminal napišemo **sudo apt-get update**, da dobimo preko interneta najnovejšo različico java, ki jo v nadeljevanju naložimo. To naredimo, da napišemo **sudo apt-get install oracle-java8-jdk**. Tako se nam bo namestila najnovejša verzija java.



```
pi@raspberrypi ~/JavaStevila $ sudo apt-get install oracle-java8-jdk
Reading package lists... Done
Building dependency tree
Reading state information... Done
oracle-java8-jdk is already the newest version.
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
```

Slika 7: Inštalacije java

Programe, ki so v jeziku java in python, lahko pišemo na računalniku in nato prenesemo na RPi, ne da bi kaj spremenili. Pri C#-pu pa to velikokrat ni mogoče, sploh pri grafičnih aplikacijah. To je zaradi tega, ker se uporablja ista knjižnica na vseh platformah, pri C# pa je ta uporaba okoli 80 %. Pri javi in pythonu velja enako tudi za konzolne kot okenske aplikacije. Pisanje java na RPi je dokaj zahtevno, saj ni prednaloženega programa za pisanje kode. Program je treba napisati v beležnici ali v konzolnem oknu. Zaradi istih knjižnic je po našem mnenju bolj primerno pisanje programa na računalniku, kjer lahko naložiš razvojna orodja oz. okolja za java (npr. Eclipse), s katerimi je delo lažje in hitreje opravljeno. Enako velja tudi za okenske aplikacije, saj lahko kodo iz osebnega računalnika samo prenesemo na RPi in deluje. V programu Eclipse je pisanje takšnih aplikacij zahtevno, saj ne moreš vstaviti gumba, ampak ga moraš napisati s programski kodo. Na RPi-ju ni nobenega uporabnega razvojnega orodja za java. Kot sam jezik je zelo primeren za uporabo, pisanje ni težavno, jezika pa se lahko hitro naučiš.

Če java primerjamo z C#-om in python-om, lahko rečemo, da je java zelo primeren jezik za uporabo na RPi-ju. Namestitev ni zahtevna, kot tudi ni zahteven jezik sam. Res je, da je pisanje programa na RPi-ju malce težje, vendar je s pomočjo računalnika zelo enostavno.

Za okenski program smo se odločili, da ga bomo napisali na osebnem računalniku, saj je veliko lažje kot na RPi-ju, kjer ni pravega razvojnega okolja za okenske programe. Program s klikom na gumb sešteje 2 števili.

### Sintaksa programa s števili:

```
import java.util.Random;
import java.util.Arrays;

public class stevila {

    public static void main(String[] args) {

        long startTime = System.currentTimeMillis();

        int stStevil = 100000;
        int[] poljeSt = new int[stStevil];

        Random stevilo = new Random();

        for(int i = 0; i < stStevil; i++)
        {
            poljeSt[i] = stevilo.nextInt(100);
        }

        Arrays.sort(poljeSt);

        for(int i = 0; i < stStevil; i++)
        {
            System.out.println(poljeSt[i]);
        }

        long endTime = System.currentTimeMillis();

        System.out.println("That took " + (endTime - startTime) + " milliseconds");
    }
}
```

Slika 8: Program z izpisom števil



### Sintaksa programa s stringi:

```
import java.util.Arrays;

public class barve {

    public static void main(String[] args) {

        long startTime = System.currentTimeMillis();

        for(int j = 0; j < 100000; j++)
        {
            String[] barve = { "crna", "modra", "zelena", "rumena", "bela", "oranjna" };
            Arrays.sort(barve);

            for(int i = 0; i < barve.length; i++)
            {
                System.out.println(barve[i]);
            }
        }

        long endTime = System.currentTimeMillis();

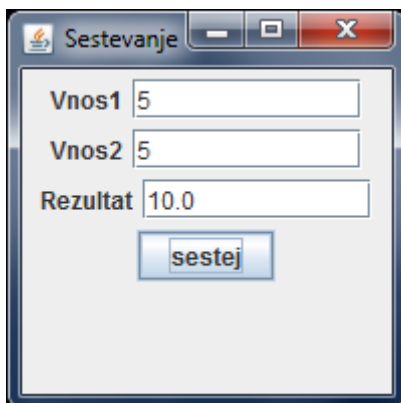
        System.out.println("That took " + (endTime - startTime) + " milliseconds");
    }
}
```

Slika 9: Slika programa z barvami

Sintaksa in izgled grafičnega programa v javi:

```
1 import java.awt.*;
2 import java.awt.event.*;
3 import javax.swing.*;
4
5 public class JavaButton extends JFrame implements ActionListener
6 {
7     JLabel l1,l2,l3;
8     JButton b1;
9     JTextField t1,t2,t3;
10
11     JavaButton()
12     {
13         l1 = new JLabel("Vnos1");
14         l2 = new JLabel("Vnos2");
15         l3 = new JLabel("Rezultat");
16         b1 = new JButton("sestej");
17         t1 = new JTextField(10);
18         t2 = new JTextField(10);
19         t3 = new JTextField(10);
20
21         add(l1); add(t1); add(l2); add(t2);
22         add(l3); add(t3); add(b1);
23
24         b1.addActionListener(this);
25         setSize(200,200);
26         setLayout(new FlowLayout());
27         setTitle("Sestevanje");
28     }
29
30     public void actionPerformed(ActionEvent e)
31     {
32         float a,b,c;
33         if(e.getSource()==b1)
34         {
35             a = Float.parseFloat(t1.getText());
36             b = Float.parseFloat(t2.getText());
37             c = a + b;
38             t3.setText(String.valueOf(c));
39         }
40     }
41
42     public static void main(String args[])
43     {
44         JavaButton a=new JavaButton();
45         a.setVisible(true);
46         a.setLocation(200,200);
47     }
48 }
49
```

Slika 10: Primer sintakse okenskega programa v javi



Slika 11: izgled okenskega programa v javi



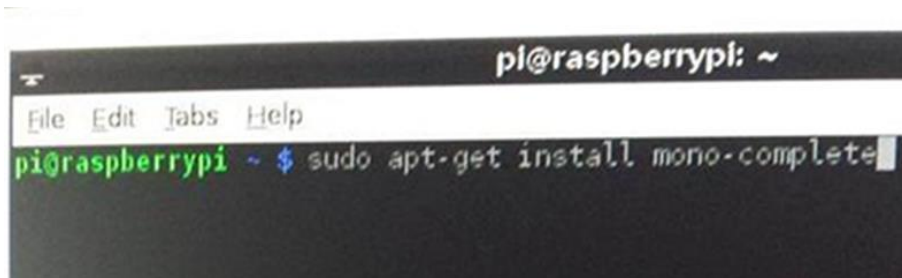
### 5.3 C#

Če smo naložili operacijski sistem Raspbian, ta že pride z nameščeno javo in pythonom, ne pa tudi z C#. Zato ga moramo namestiti posebej. To naredimo tako, da zaženemo LXTerminal.



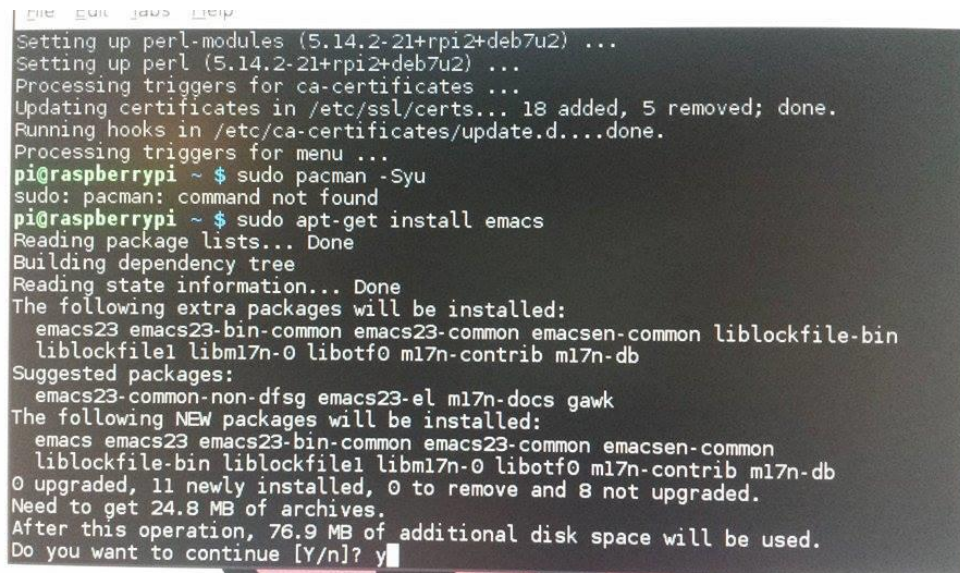
Slika 12: Zagon LXTerminal

Ko se pojavi okno LXTerminala, vpišemo komando: **sudo apt-get update**, ki bo posodobila Raspbian na najnovejšo različico. Potem ko smo posodobili, vpišemo komando: **sudo apt-get install mono-complete**, skatero se bo namestilo jedro .NET Framework, ki je potrebno za delovanje C#.



Slika 13: Okno LXTerminal

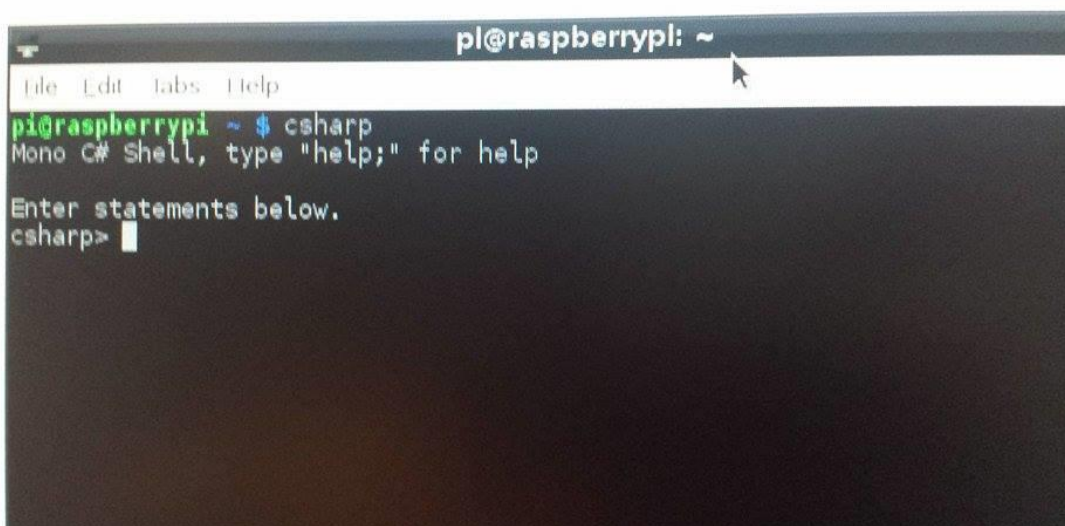
Sedaj lahko po želji izberemo urejevalnik besedil, v katerega bomo pisali kodo. Na izbiro imamo veliko urejevalnikov besedil, kot so (LEaFPad, emacs ...), izberemo pa tistega, ki nam najbolj ustreza. Mi smo izbrali LEaFPad, ki je že naložen, če pa želimo kakšen drug urejevalnik besedil (npr. emacs), pa uporabimo komando **sudo apt-get install emacs** in počakamo, da se prenese.



```
pi@raspberrypi ~$ sudo pacman -Syu
Setting up perl-modules (5.14.2-21+rpi2+deb7u2) ...
Setting up perl (5.14.2-21+rpi2+deb7u2) ...
Processing triggers for ca-certificates ...
Updating certificates in /etc/ssl/certs... 18 added, 5 removed; done.
Running hooks in /etc/ca-certificates/update.d....done.
Processing triggers for menu ...
pi@raspberrypi ~$ sudo pacman -Syu
sudo: pacman: command not found
pi@raspberrypi ~$ sudo apt-get install emacs
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following extra packages will be installed:
  emacs23 emacs23-bin-common emacs23-common emacs23-el emacs23-el-common emacs23-el-common
  libblockfile1 libm17n-0 libotf0 m17n-contrib m17n-db
Suggested packages:
  emacs23-common-non-dfsg emacs23-el m17n-docs gawk
The following NEW packages will be installed:
  emacs emacs23 emacs23-bin-common emacs23-common emacs23-el emacs23-el-common
  libblockfile1 libblockfile1-bin libm17n-0 libotf0 m17n-contrib m17n-db
0 upgraded, 11 newly installed, 0 to remove and 8 not upgraded.
Need to get 24.8 MB of archives.
After this operation, 76.9 MB of additional disk space will be used.
Do you want to continue [Y/n]? y
```

Slika 14: Nameščanje emacs

Po končanem prenosu smo preverili, če smo pravilno namestili .NET Framework in C#, to pa smo storili tako, da smo v LXTerminal vpisali **csharp**, in je delovalo kot prikazano na spodnji sliki:

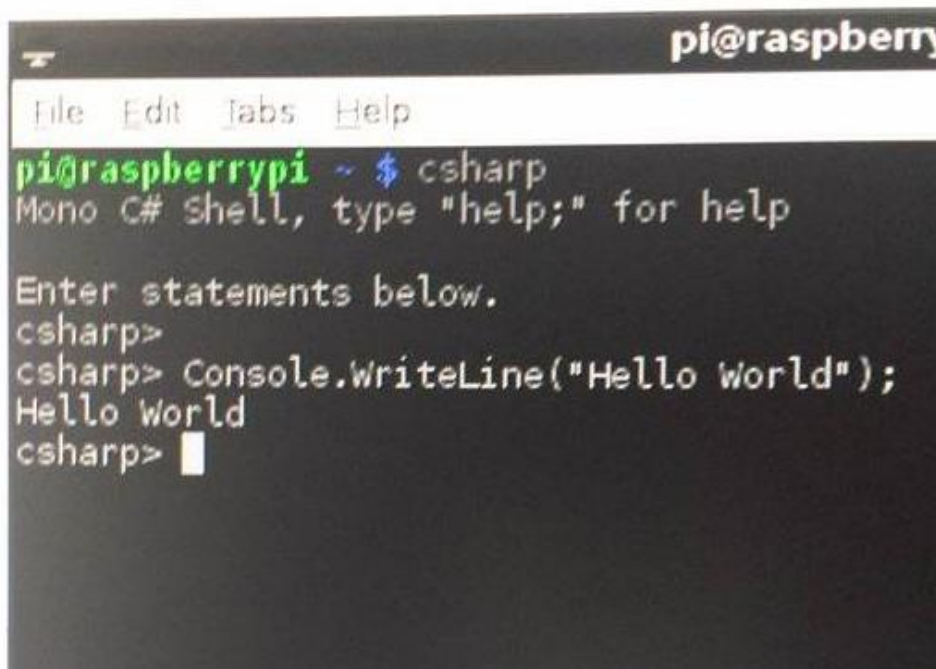


```
pi@raspberrypi ~$ csharp
Mono C# Shell, type "help;" for help

Enter statements below.
csharp>
```

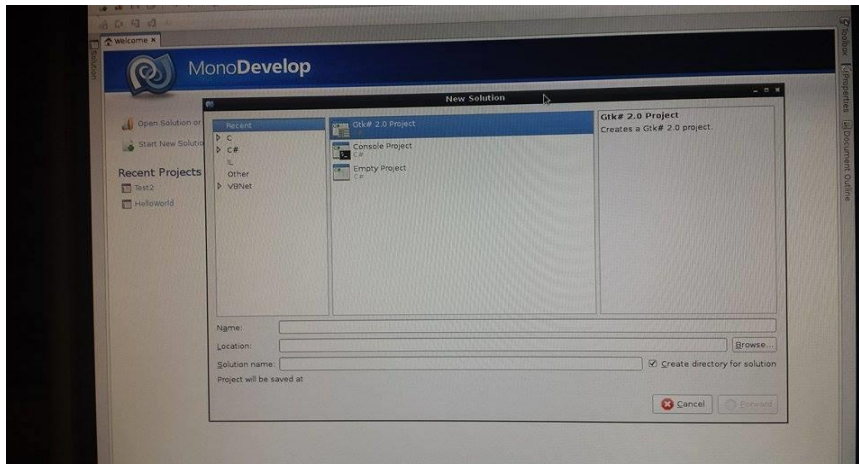
Slika 15: C# v konzoli

Nato smo testirali še lažji program, da smo se prepričali, če deluje tako, kot je treba. Pravilno delovanje programa je prikazano na spodnji sliki. Ko smo končali z uporabo C#, napišemo samo komando **quit**, s čimer pridemo nazaj v LXTerminal.

A screenshot of a terminal window titled 'pi@raspberrypi'. The window has a menu bar with 'File', 'Edit', 'Tabs', and 'Help'. The terminal shows the following text: 'pi@raspberrypi ~ \$ csharp', 'Mono C# Shell, type "help;" for help', 'Enter statements below.', 'csharp>', 'csharp> Console.WriteLine("Hello World");', 'Hello World', and 'csharp>' followed by a cursor. The text is displayed in a monospaced font with some color coding (green for the prompt, blue for the command, and white for the output).

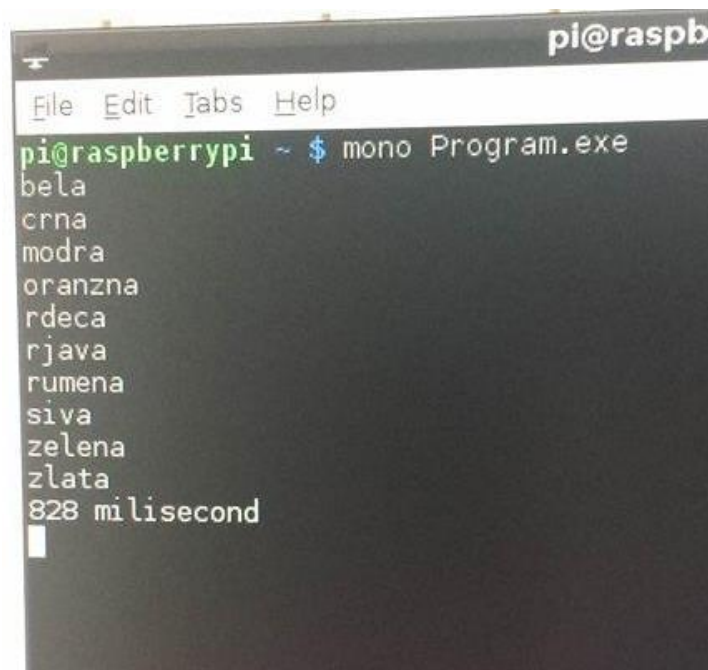
Slika 16: Osnovni program v C#

Sprva smo hoteli uporabljati MonoDevelop, ki je zelo grafično izpopolnjeno orodje za pisanje konzolnih programov kot grafičnih. Je zelo podoben Visual Studiu, zato se ga kar hitro navadiš. Je tudi primerna izbira za manj izkušene programerje, saj je na voljo več sredstev, s katerimi si lahko pomagamo. Med temi sredstvi bi posebej izpostavili dopolnjevanje ukazov, pokažejo pa se tudi logične in skladenske napake. Čeprav je zelo močno orodje, smo naleteli na nekaj težav pri njegovi uporabi, saj je delal izredno počasi ter je med uporabo dvakrat celo prenehal delovati. Na tej točki smo se odločili, da bomo delo nadaljevali s programom LeafPad, saj je hitrejši in zanesljivejši. Slabost pri uporabi tega programa pa je, da je potrebno ročno vpisati vse knjižice, ki jih želimo uporabljati, nima nikakršne podpore za samo izpolnjevanje ukazov, prav tako pa ne pokaže napak ali logičnih problemov v kodi.



Slika 17: MonoDevelop

Omenjeni program smo izvajali kar preko LXTerminala. Najprej smo ga morali »compilati« s komando **mcs (Ime datoteke).cs**, ta komanda je nato ustvarila datoteko exe. Nato smo v LXTerminalu uporabili komando **mono (ime datoteke).exe** in program se je izvedel, kot je razvidno s slike st. 18.



Slika 18: Izvajanje programa v LXTerminal



```

Edit Search Options Help
using System;
using System.Collections.Generic;
using System.Diagnostics;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace SharpString
{
    class Program
    {
        static void Main(string[] args)
        {
            Stopwatch stopwatch = Stopwatch.StartNew();
            stopwatch.Start();

            String[] barve = { "crna", "modra", "zelena", "rumena", "bela" };
            Array.Sort(barve);

            for(int i = 0; i < barve.Length; i++)
            {
                Console.WriteLine(barve[i]);
            }

            stopwatch.Stop();
            Console.WriteLine(stopwatch.ElapsedMilliseconds + " milisecon");

            Console.ReadKey(true);
        }
    }
}

```

Slika 19: Koda v LeafPadu

Alternativa pa je tudi, da napišemo konzolski program na PC-ju. To zelo priporočamo, saj na PC-ju hitreje napišeš program in hkrati so ti ponujena v uporabo različna orodja, med katerimi lahko izbiramo in s katerimi si olajšamo delo. Nato ga prenesemo na RPi . Naredili smo dva konzolna programa, program z naključnimi števili 1 - 100 in ga dali v zanko, ki je to izvedla 100 000-krat. Program si je shranjeval števila v polje, kot je razvidno na sliki 20. Drugi program pa smo izdelali tako, da je najprej kopiral stringe v novo polje in jih nato razvrstil po abecedi ter jih nato izpisal 100 000-krat (slika 21). V vsakem od programov smo uporabili še kodo time, da smo izmerili čas izvajanja programa tako na PC-ju kot na RPi-ju.

Primeri dveh konzolnih programov:

```
using System;
using System.Collections.Generic;
using System.Diagnostics;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace SharpSt
{
    class Program
    {
        static void Main(string[] args)
        {
            Stopwatch stopwatch = Stopwatch.StartNew();
            stopwatch.Start();

            int stStevil = 100000;
            int[] poljeSt = new int[stStevil];

            Random stevilo = new Random();

            for (int i = 0; i < stStevil; i++)
            {
                poljeSt[i] = stevilo.Next(0, 100);
            }

            Array.Sort(poljeSt);

            for (int i = 0; i < stStevil; i++)
            {
                Console.WriteLine(poljeSt[i]);
            }

            stopwatch.Stop();
            Console.WriteLine(stopwatch.ElapsedMilliseconds + " milisecond");

            Console.ReadKey(true);
        }
    }
}
```

Slika 20: Program z generiranjem naključnih števil

```

using System;
using System.Collections.Generic;
using System.Diagnostics;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace SharpString
{
    class Program
    {
        static void Main(string[] args)
        {
            Stopwatch stopwatch = Stopwatch.StartNew();
            stopwatch.Start();

            for (int j = 0; j < 100000; j++)
            {
                String[] barve = { "crna", "modra", "zelena", "rumena", "bela" };

                Array.Sort(barve);

                /*
                for (int i = 0; i < barve.Length; i++)
                {
                    Console.WriteLine(barve[i]);
                }
                */

                stopwatch.Stop();
                Console.WriteLine(stopwatch.ElapsedMilliseconds + " milisecond");

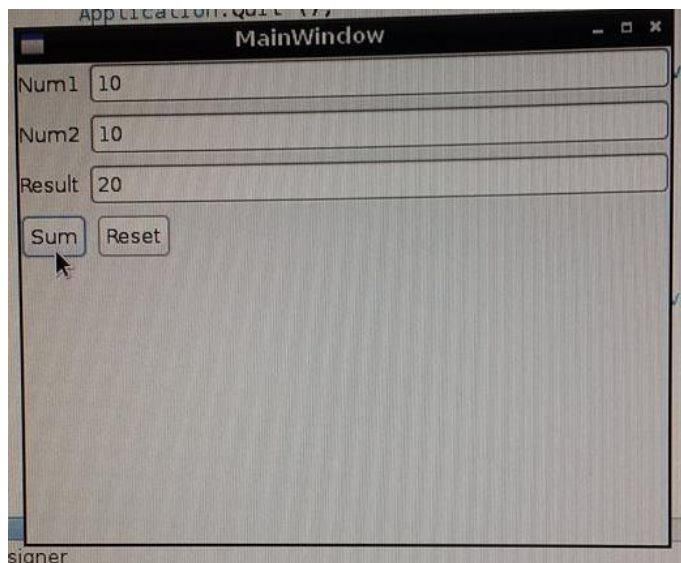
                Console.ReadKey(true);
            }
        }
    }
}

```

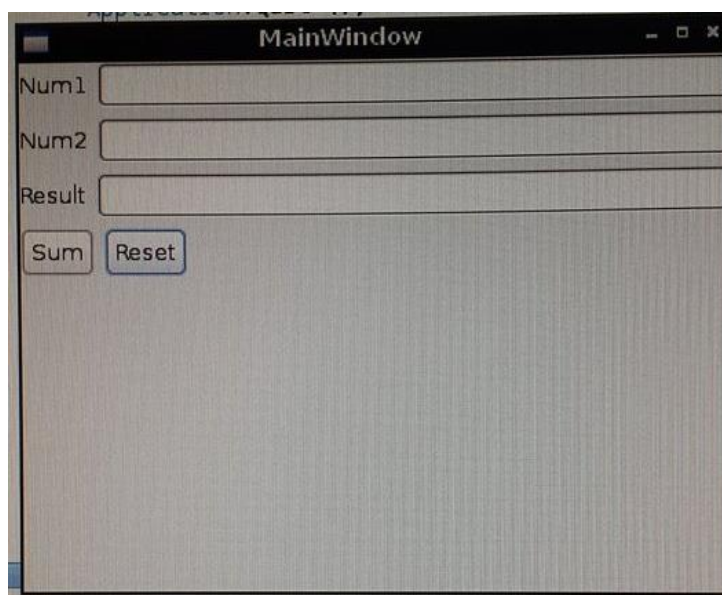
Slika 21: Razvršćanje barv po abecedi

Za grafični program smo se odločili izdelati preprost kalkulator v razvojnem okolju MonoDevelop, za seštevanje dveh števil, katerih rezultat se nato izpiše. Imamo pa tudi dva gumba in sicer enega za seštevanje, drugega pa za izbris podatkov.

Primer grafičnega programa:



Slika 22: Grafični program - kalkulator – 1 C#



Slika 23: Grafični program - kalkulator – 2 C#



```

w> OnButton2Clicked (object sender, EventArgs e)

public partial class MainWindow: Gtk.Window
{
    Build ();
}

protected void OnDeleteEvent (object sender, DeleteEventArgs a)
{
    Application.Quit ();
    a.RetVal = true;
}

protected void OnButton1Clicked (object sender, EventArgs e)
{
    int a,b,c;

    a=Convert.ToInt32(num1.Text);
    b=Convert.ToInt32(num2.Text);
    c=a+b;
    result.Text=c.ToString();
}

protected void OnButton2Clicked (object sender, EventArgs e)
{
    num1.Text=num2.Text=result.Text="";
}

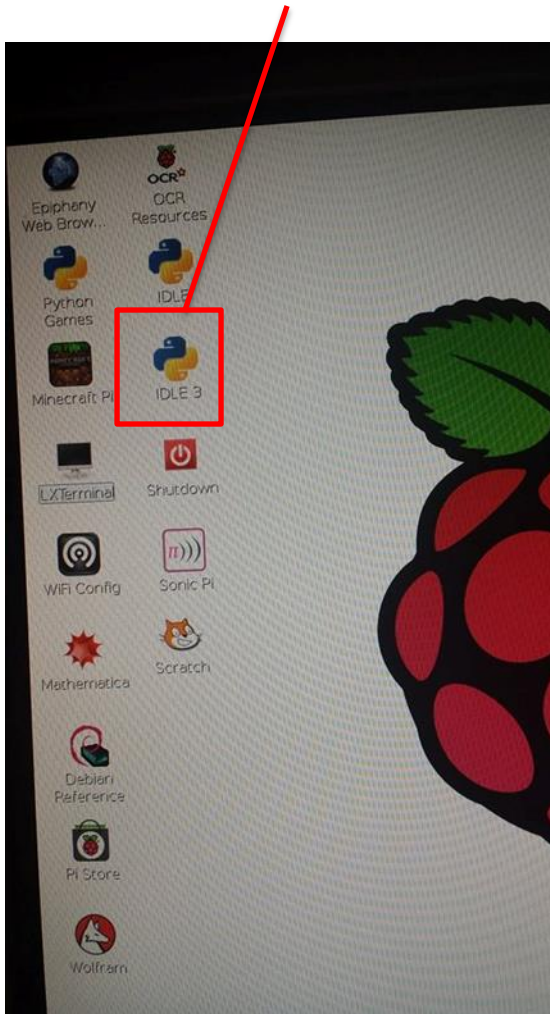
```

Slika 24: Sintaksa grafičnega programa C#

Naredili smo tudi grafični program na osebnem računalniku v visual studiu ter ga nato prenesli na RPi-ja. Pretvorili smo ga v konzoli v .exe zapis in ga nato zagnali, ampak ni deloval. Težavo smo poskušali rešiti z različnimi metodami, ki pa tudi niso delovale, zato smo program naredili kar v razvojnem okolju MonoDevelop na RPi, v katerem nam je vzelo eno uro, da smo napisali preprost kalkulator. Če pišemo konzolske programe, je po naših izkušnjah primerneje napisati program na osebnem računalniku in ga nato samo prenesti na RPi-ja.

## 5.4 Python

Python je privzeti programski jezik na RPi. Na starejših NOOBS-ih je naložen python 2, na novejših pa sta naložena python 2 in python 3. Je najenostavnejši programski jezik izmed teh ki smo jih uporabili. Python na Rapsberry Pi lahko zaženemo z namizja tako, da dvakrat kliknemo na ikono IDLE 3 (*prikazano na spodnji sliki*).



Slika 25: Ikona za zagon IDLE 3

Kot osnovni program smo naredili program, ki izpiše »Hello World«:

```
>>> print ("hello world")
hello world
>>>
```

Slika 26: Testni program python

Nato smo naredili tudi dva nekoliko zahtevnejša programa. Tako kot pri javi in C# smo naredili enega za naključno izbiro števil 1 - 100 in polje stringov z desetimi barvami. Napisali smo program, ki je generiral 100.000 celih števil od 0 do 100 in jih shranil v polje, ga sortiral in na koncu izpisal.

Sintaksa programa s števili:

```
import time
import random
start_time = time.clock()

stevila = []

for i in range(0, 100000):
    num = random.randint(0, 100)
    stevila.append(num)

stevila.sort()

for s in stevila:
    print (s)
print

print (time.clock() - start_time)
```

Slika 27: Program za izpis in razvrščanje števil

Nato smo izdelali tudi program, ki polje stringov dolžine 10 sortira in to ponovi 100.000-krat. Za vhodne podatke smo izbrali kar imena barv.

Sintaksa programa s stringi:

```
import time

start_time = time.clock()

for i in range (1, 100000):
    barve = ["crna", "modra", "zelena", "rumena", "bela"]
    barve.sort()

    for b in barve:
        print (b)
    print

print (time.clock() - start_time)
```

Slika 28: Program z razvrščanjem števil

Izdelali smo tudi grafični program, ki sešteje dve števili. To lahko storimo z različnimi orodji, kot so Tkinter, wxPython, IronPython ... Mi smo uporabili Tkinter, ker lahko zanj knjižnico preprosto uvozimo z ukazom **import tkinter**.

Sintaksa in izgled grafičnega programa:

```
from tkinter import *

top = Tk()

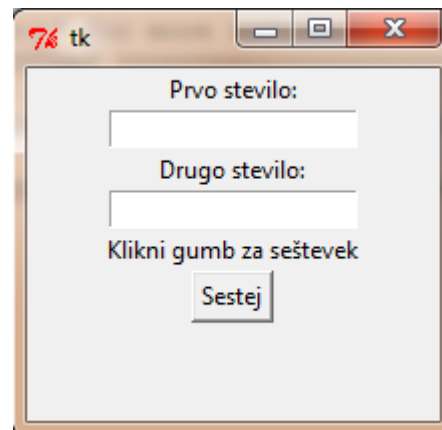
resultStr= StringVar()
resultStr.set("Klikni gumb za seštevek")
number1Label = Label (text="Prvo stevilo:")
number1Label.pack()
number1Box = Entry()
number1Box.pack()

number2Label = Label (text="Drugo stevilo:")
number2Label.pack()
number2Box = Entry()
number2Box.pack()

resultLabel = Label(textvariable=resultStr)
resultLabel.pack()

def addNo():
    no1 = int(number1Box.get())
    no2 = int(number2Box.get())
    no3 = no1+no2
    resultStr.set( "Seštevek je: "+str(no3))
but = Button(text="Sestej", command=addNo)
but.pack()

top.mainloop()
```



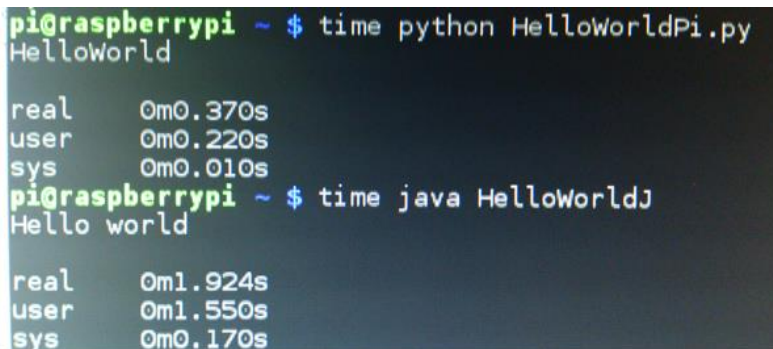
Slika 29: Izgled grafičnega programa

Slika 30: Sintaksa grafičnega programa python

Izdelava grafičnega programa v tkinterju je dokaj enostavna. Nekoliko zamudno je lahko samo učenje sintakse, vendar nam to ni predstavljalo večjih težav. Razvojno okolje je dokaj enostavno, saj je podobno osnovnemu pythonu. Manjši problem predstavlja le, da sproti ne moremo preverjati, kako stojijo elementi v naši aplikaciji in moramo zato pogosto zaganjati aplikacijo.

## 6 Testiranje

Testirali smo, v kolikšnem času se izvedejo enaki programi v različnih programskih jezikih. Za osnovni program smo izpisali »Hello World«.



```
pi@raspberrypi ~ $ time python HelloWorldPi.py
HelloWorld
real    0m0.370s
user    0m0.220s
sys     0m0.010s
pi@raspberrypi ~ $ time java HelloWorldJ
Hello world
real    0m1.924s
user    0m1.550s
sys     0m0.170s
```

Slika 31: Hello World v python in java

Prvi rezultati so zanimivi. V pythonu se je program izvedel v 0,370 s, v javi pa v »dolgih« 1,924 s.

Ukaz time je močan ukaz, vendar je za izračun časa izvajanja nekega programa veliko bolj natančno, če naredimo »timer« v programu samem.

Program 1 je program, ki dela s števili, program 2 pa dela s stringi. Manjša, kot je vrednost časa, boljši je rezultat. Odločili smo se tudi oba programa testirati brez izpisa števil ter besed, da ugotovimo kje se programi bolje obnesejo.

Vse teste smo opravljali na istem osebnem računalniku, da zagotovimo primerljive podatke. Specifikacije so:

- Grafična kartica: ATI Radeon 3000 Graphics
  - Procesor: AMD Phenom™ II X4 830 Processor 3.01 GHz
  - RAM: 4GB DDR3 1333 MHz
  - Operacijski sistem: Windows 8.1 Enterprise
- disk: Hitachi HDS721050CLA362 ATA 500 GB

V tem delu bomo prikazali rezultate postopka dela naše raziskovalne naloge. Za prikaz teh podatkov smo uporabili tabele in grafe:

**Tabela 2: Čas izvajanja java**

java	PC	RPi
Program 1 (števila)	0,90 s	32,80 s
Program 2 (stringi)	7,89 s	285,80 s
Program 1 (števila, brez izpisa)	0,02 s	0,32 s
Program 2 (stringi, brez izpisa)	0,08 s	1,16 s

**Tabela 3: Čas izvajanja C#**

C#	PC	RPi
Program 1 (števila)	10,47 s	24,66 s
Program 2 (stringi)	100,20 s	366,39 s
Program 1 (števila, brez izpisa)	0,01 s	0,33 s
Program 2 (stringi, brez izpisa)	0,48 s	41,68 s

**Tabela 4: Čas izvajanja Phytton**

python	PC	RPi
Program 1 (števila)	14,22 s	12,41s
Program 2 (stringi)	214,51 s	56,54 s
Program 1 (števila, brez izpisa)	0,27 s	6,02 s
Program 2 (stringi, brez izpisa)	0,12 s	2,12 s

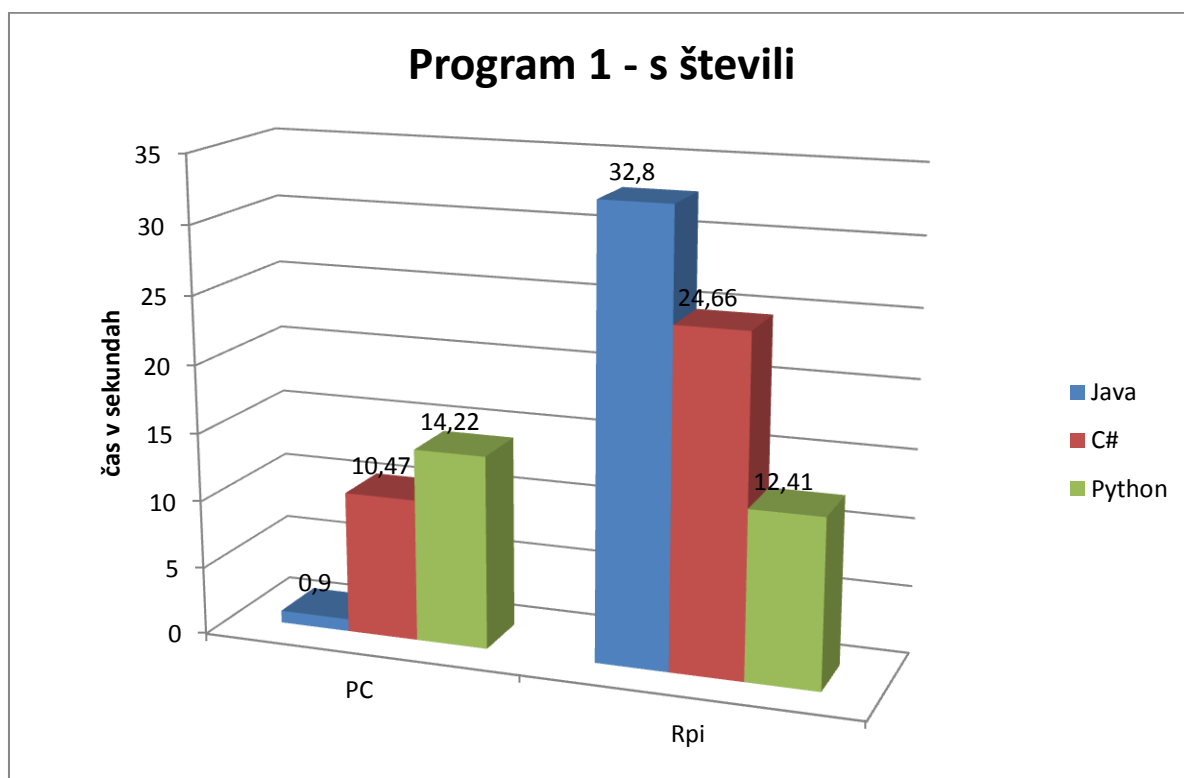
**Tabela 5: Čas izvajanja na Raspberry Pi**

RPi	java	C#	python
Program 1 (števila)	30,80 s	24,66 s	12,41s
Program 2 (stringi)	285,80 s	366,39 s	56,54 s
Program 1 (števila, brez izpisa)	0,32 s	0,33 s	6,02 s
Program 2 (stringi, brez izpisa)	1,16 s	41,68 s	2,12 s

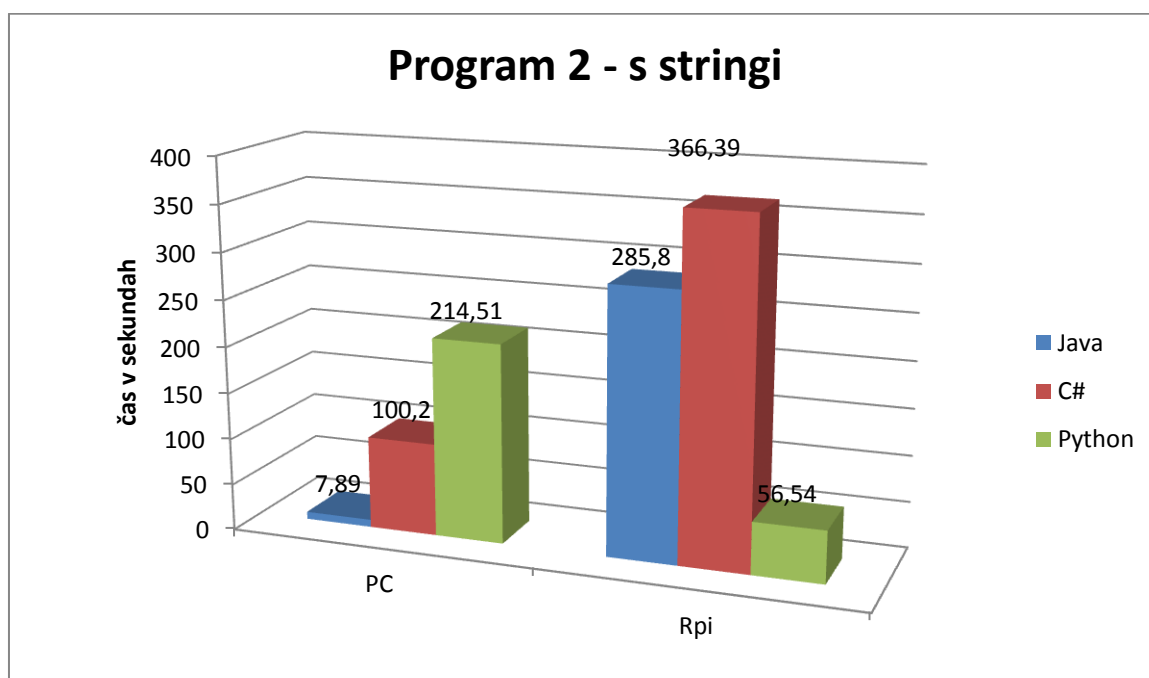
**Tabela 6: Čas izvajanja na PCju**

PC	java	C#	python
Program 1 (števila)	0,90 s	10,47 s	14,22 s
Program 2 (stringi)	7,89 s	100,20 s	214,51 s
Program 1 (števila, brez izpisa)	0,02 s	0,01 s	0,27 s
Program 2 (stringi, brez izpisa)	0,08 s	0,48 s	0,12 s

Grafi:



Graf 1: Grafični prikaz časa izvajanja za program 1



Graf 2: Grafični prikaz časa izvajanja za program 2

## 7 Analiza rezultatov

Po končanih testih na Raspberry Pi nismo bili presenečeni. Po pričakovanjih sta bila najhitrejša java in python, vendar je pri pythonu pogojeno s tem, ali smo uporabili ukaz za izpis ali ne, saj nas je zelo presenetilo to, da je bil python hitrejši takrat ko smo uporabili izpis števil ali stringov. Izkazalo se je, da je java najboljša, če vzamemo za primerjavo izvedbo programov brez izpisa. C# se je izkazal po pričakovanjih, vendar je pri izvedbi programa brez izpisa stringov delal presenetljivo počasi, kar tudi sami nismo znali razložiti. Problemi so se tudi pojavili pri C#-ovem v razvojnem okolju MonoDevelop, pri katerem je bila največja težava počasno delovanje samega razvojnega okolja. Najmanj težav se je pojavilo prav pri tretjem programskem jeziku – javi. Ta je večino časa deloval brezhibno in z njim ni bilo nobenih večjih težav. Nad rezultati smo bili presenečeni, saj je že pri samem C# pri programu št. 2 brez izpisa prišlo do zelo velike razlike. Java nas je tudi zelo pozitivno presenetila, saj se je pri testiranju zelo dobro izkazala. Prišli smo do sklepa, da ima vsak programski jezik svoje karakteristike za določene dele zgradbe sintakse in hitrost samega izvajanja programa ali izpisa tega programa. To pomeni, da na primer v našem primeru ima python veliko enostavnejšo sintakso za izpis kot jezika C# in java, vendar ima veliko počasnejše branje programa. Zato smo tudi tukaj prišli do sklepa, da je bolje uporabiti python, kadar želimo izpisati podatke, ki smo jih uporabili v programu.

Pri izdelavi te raziskovalne naloge smo se naučili, da je pri C# na RPi razvojno okolje MonoDevelop zelo prijazno uporabnikom, saj je podoben Visual Studiu, ki ga uporabljamo v šoli, vendar njegove uporabe še ne priporočamo, saj je delovanje zelo počasno in nestabilno. Zato predlagamo, da za program v C# uporabite Leafpad ali pa še bolje, da ga uvozite z računalnika. Programski jezik python, ki se je odlično izkazal pri naših testih, je zelo dober za tiste, ki želijo z manj kode napisati več. Naš tretji jezik – java pa tudi močno priporočamo, saj je zelo prilagodljiva in se je tudi dobro izkazala pri našem testiranju. Prav tako je najbolj razširjen programski jezik na svetu. Z njim smo imeli najmanj problemov, kljub temu da je težko v javi izdelovati okenske oziroma grafične aplikacije. Na podlagi naših izkušenj in testiranj smo prišli do skupnega zaključka, da je java najboljše razvojno okolje izmed testiranih. Vendar moramo upoštevati, da python deluje veliko hitreje od jave in C# kadar v aplikaciji uporabimo izpis podatkov.

Preden zaključimo z našo raziskovalno nalogo, bomo na kratko zapisali še, ali smo potrdili ali ovrgli naše hipoteze.

- **Najprimernejši jezik za Raspberry Pi je python.**

Python je najbolj primeren programski jezik za izdelavo programov na Raspberry Pi, čeprav so testi pokazali, da je podobno hiter kot java in hitrejši od C #, vendar samo v primeru, da ne uporabimo izpisov, saj se je v tem primeru python izkazal za nekoliko slabšega. Zato smo



prišli do sklepa, da je python zelo primeren jezik, saj je že naložen na RPi-ju. Vendar se nam tudi poda vprašanje, ali se bodo začetniki res učili programiranja na Raspberry Pi? Velika verjetnost je, da se bo vsak najprej začel učiti na namiznem računalniku, saj so na njem načini učenja same zgradbe jezikov in programiranja skoraj neomejeni.

- **Raspberry Pi lahko v prihodnosti konkurira osebnim računalnikom.**

Ta hipoteza drži, če zraven upoštevamo še ceno ter velikost. Vsekakor bo RPi v prihodnosti postal še velika konkurenca računalnikom. To se na trenutnem trgu že na nek način dogaja, saj je v začetku februarja prišel novi Raspberry Pi 2. Na nek način postaja Raspberry Pi konkurenca našim računalnikom, saj je prvi prišel na trg okoli leta 2012 in se pričakovano hitreje razvija kot računalniška oprema. Na nek način se Raspberry Pi zgleduje po njej in lahko na nek način »uvozi« tehnologije strojne in programske opreme navadnih računalnikov.

- **C# je primeren za delo na Raspberry Pi.**

Kot programski jezik na RPi se je C# izkazal zelo obetavno za nadaljnji razvoj. V našem primeru smo naleteli na kar nekaj težav z razvijalskim okoljem MonoDevelop, saj je nekoliko slabo prilagojen za razvoj okenskih aplikacij, pri ustvarjanju konzolskih programov pa je zelo uporaben na RPi. V bližnji prihodnosti lahko zato pričakujemo, da bo MonoDevelop na RPi postal še bolj prilagojen. Bližnja prihodnost je zelo obetavna, saj naj bi Raspberry Pi 2 imel več procesorske moči ter omogočal prihajajoči Windows 10, kar pomeni, da lahko postaneta C# in MonoDevelop nekoliko bolj uporabljena pri samem razvoju aplikacij na Raspberry Pi 2.

- **Java učinkovita pri delu na Raspberry Pi.**

Kot so pokazali naši testi, je java zelo učinkovita pri delu na Raspberry Pi. Čeprav je nekoliko težja izdelava okenskih aplikacij, lahko tudi mi potrdimo, da na splošno drži, kar pravijo drugi, da je java vsestranski jezik. Zaradi svoje prilagodljivosti nam omogoča, da lahko programe prenašamo z ene platforme na drugo dokaj enostavno v primerjavi z nekaterimi jeziki.

- **Programi na Raspberry Pi se hitreje izvajajo kot na osebnem računalniku.**

Kot so pokazali naši testi, je bilo verjetno pričakovano, da je hitreje izvedel programe namizni računalnik. Vzrok je ta, da ima Raspberry Pi eno samo jedro in manjšo frekvenco kot namizni računalnik. Le-ti pa so hitrejši še zaradi enega vzroka, in sicer da lahko PC nameni izvajanju testnega programa več jeder kar avtomatsko pomeni, da je hitrejši. Vendar nas je pozitivno presenetilo to, da se je program 1 z izpisom hitreje izvedel na Raspberry Pi kot na PC-ju.

- **Raspberry Pi je primeren pripomoček za izvajanje poučevanja programiranja.**

Raspberry Pi je vsekakor primeren predvsem za učenje osnov programiranja zaradi svoje prenosljivosti in enostavnosti uporabniškega vmesnika. Omeniti velja, da ima že vnaprej naložen grafični programski jezik Scratch, ki je predvsem primeren za osnovnošolce. V Scratchu je programiranje zelo enostavno, saj ni potrebno pisati kode, ker preprosto klikamo gumbе in jezik sam generira kodo. Edini problem je lahko krhkost in izpostavljenost vseh komponent. To manjšo težavo pa lahko odpravimo z morebitnim nakupom zaščitne plastike, ki jo lahko kupimo poleg Raspberry Pi-a.

- **Na Raspberry Pi se splača izdelovati grafične aplikacije.**

Izdelovanje grafičnih aplikacij se na RPi skorajda ne splača. Razlog za to je med drugim tudi to, da je dosti razvijalskih okolij slabo prilagojenih RPi. Mi smo uporabljali MonoDevelop, ki je vmesnik za C#. Ta se je izkazal za zelo počasnega in zamudnega, med drugim pa se je kar nekajkrat sesul. Kljub temu da se je slabo izkazal na naših testiranjih, pa se je izkazal za obetavnega. Verjetno se MonoDevelop in drugim razvijalskim orodjem obetajo boljši časi, če bodo smernice njihovega razvoja šle v pravo smer. Za ostala programska jezika pa za RPi ni nobenega uporabnega orodja za razvoj grafičnih aplikacij.

## **Zaključek**

Kot so pokazali naši rezultati, je najprimernejši programski jezik na Raspberry Pi python, ki se je najbolje izkazal pri našem testiranju. Nasploh pa je python znan kot zelo prilagodljiv in uporaben programski jezik za katerega velja da lahko z manj kode naredimo več. Zelo uporabljen je ravno zaradi svoje prilagodljivosti in tudi zato, ker je zgradba oziroma sintaksa tega jezika zelo učinkovita ter se izvaja hitreje v primerjavi z ostalimi jeziki kadar uporabimo izpis. Pozitivno nas je presenetil C#, ki se je kljub nekoliko neodzivnemu, ampak zelo uporabnemu razvojnemu okolju MonoDevelop, izkazal za dobro izbiro, kljub temu da smo aplikacije v tem jeziku pisali na osebнем računalniku v Visual Studiu in jih potem samo prenesli na RPi. Za java pa velja omeniti, da je programe izvedla najhitreje kadar nismo uporabili izpisa polja.

Na koncu velja omeniti tudi, da je v času izdelave naše raziskovalne naloge na trg v začetku februarja nepričakovano prišel Raspberry Pi 2, ki je še zmogljivejši od svojega predhodnika in z boljšo podporo za razvojna okolja. Med drugim so govorice, da naj bi omogočal podporo prihajajočemu operacijskemu sistemu Windows 10, kar pomeni, da bo torej omogočal vse njegove knjižnice, razvojno okolje C# in tudi .NET Framework, ki smo si ga naložili na začetku testiranja.

Z izidom novega Raspberry Pi smo pridobili smernice za nadaljnje delo z miniaturnimi računalniki. Lotili se bomo testiranja Raspberry Pi 2 in ga nato primerjali z verzijo B+, ki jo uporabljamo trenutno mi. Primerjali ju bomo ,kateri deluje hitreje in predvsem, kaj jim je uspelo izboljšati od predhodnika. Mogoče se bomo lotili tudi katerega drugega takšnega računalnika, npr. Banana Pi. Tehnologija je v zadnjih letih močno napredovala, med drugim tudi na področju miniaturnih računalnikov. Zato lahko v naslednjih nekaj letih pričakujemo, da se bodo še nekoliko bolj razširili in morda v prihodnosti zamenjali namizne računalnike.

## **Zahvala**

Zahvaljujemo se našemu mentorju, profesorju mag. Boštjanu Resinoviču, za pomoč pri izdelavi raziskovalne naloge, ter za začasno izposojlo opreme (Raspberry Pi).

Zahvaljujemo se tudi lektorici, gospe Aniti Laznik, ki je naše delo jezikovno pregledala.

Zahvalili bi se radi tudi Klavdiji Špur Jereb za pregled povzetka v angleškem jeziku.

## Viri

Bézine, E. (23. oktober 2013). *Using C# Interactive Shell*. Prevezeto 5. marec 2015 iz Raspberry#: <http://www.raspberry-sharp.org/>

Bijan. (15. junij 2015). *Python vs. Java performance*. Prevezeto 8. marec 2015 iz Stackoverflow: <http://stackoverflow.com/questions/3044620/python-vs-java-performance-runtime-speed>

*Comparing Python to Other Languages*. (brez datuma). Prevezeto 8. marec 2015 iz python: <https://www.python.org/doc/essays/comparisons/>

MADHUSUDANAN, A. (16. april 2013). *Hack Raspberry Pi – How To Build Apps In C#, WinForms and ASP.NET Using Mono In Pi*. Prevezeto 3. marec 2015 iz Amazedsaint's Tech Journal: <http://www.amazedsaint.com/2013/04/hack-raspberry-pi-how-to-build.html>

melikescheese. (6. junij 2012). *MonoDevelop C# on RPI (or Visual Studio for Linux)*. Prevezeto 5. marec 2015 iz raspberrypi forums: <http://www.raspberrypi.org/forums/viewtopic.php?f=34&t=7631>

*Raspberry Pi*. (Marec 2015). Pridobljeno iz <http://www.raspberrypi.org/>

*RPi Java*. (9. april 2013). Prevezeto 6. marec 2015 iz Embedded Linux Wiki: [http://elinux.org/RPi\\_Java](http://elinux.org/RPi_Java)

Sims, G. (23. junij 2014). *How to Install the Java Development Kit on a Raspberry Pi*. Prevezeto 2. marec 2015 iz maketecheasier: <http://www.maketecheasier.com/install-java-development-kit-raspberry-pi/>