KLIN predmet, Elektrotehnički fakultet Univerziteta u Beogradu

Merenje signala (Arduino & Pajton)

Dr Nadica Miljković, vanredni profesor

Deo materijala prikazan u ovoj prezentaciji odgovara planu i programu predmeta Merni sistemi u računarstvu (13E053MSR, http://automatika.etf.rs/sr/13e053msr), pa su i korišćeni materijali sa ovog predmeta za tekuću prezentaciju.

ARDUINO PROGRAMIRANJE

Arduino softver

- Arduino softver otvorenog koda je napisan u Java, C i C++. Operativni sistemi na kojima je moguće instalirati Arduino softver i programirati u njemu su: Windows, macOS i Linux, https://en.wikipedia.org/wiki/Arduino.
  - Arduino je veoma sličan C++ programskom jeziku.
- NAPOMENA: Nisu sve funkcije dostupne za UNO R3 hardver koji se koristi na vežbama. Neke funkcije iako postoje u softveru su specifične za odgovarajući hardver. Potrebno je pogledati uputstvo za funkciju koja se koristi.
- Slika: By Unknown author - http://arduino.cc, Public Domain, https://commons.wikimedia.org/w/index.php?curid=24588192

Promenljive. Kako ih definisati?

```
int ledPin = 8;
const int cledPin = 8;
```

- Ako su globalne:
  - Pre *setup()* funkcije. Mogu i pre *loop()* funkcije, ako se ne pozivaju u *setup()* funkciji.
  - Na slici je dato za celobrojne vrednosti (broj digitalnog pina).
  - Uobičajeno bi konstante trebalo da počinju slovom “c” kako je i prikazano na slici.
- Ako su lokalne:
  - Unutar funkcije (*setup()*, *loop()* ili neke druge koju korisnik definiše)

Kontrolne strukture (tok programa)

Control Structures

- if
- if...else
- for
- switch case
- while
- do... while
- break
- continue
- return
- goto

- Kontrolne strukture su prikazani u tabeli.
- Detalji se mogu pogledati na: https://www.arduino.cc/en/Reference/HomePage.

if struktura

```
int ledPin = 8;
void setup() {
  if (ledPin < 0) {
    ledPin = 13;
  }
  pinMode(ledPin, OUTPUT);
}
```

- Prikazana je na slici.
- Početak i kraj se označavaju vitičastim zagradama “{}”.

if else struktura

```
if (pinLed >= 13) {
  delay(200);
}
else {
  delay(800);
}
```

- Prikazana je na slici.
- Početak i kraj se označavaju vitičastim zagradama “{}”.
- Automatsko popunjavanje koda: kada se ukuca početak strukture tj. “{” i pritisne taster ENTER za prelazak u novi red, onda se automatski generiše i kraj strukture tj. “}”.

switch struktura

```
switch (range) {
  case 0:    // your hand is on the sensor
    Serial.println("dark");
    break;
  case 1:    // your hand is close to the sensor
    Serial.println("dim");
    break;
  case 2:    // your hand is a few inches from the sensor
    Serial.println("medium");
    break;
  case 3:    // your hand is nowhere near the sensor
    Serial.println("bright");
    break;
}
```

- Primer korišćenja ove strukture je prikazan na slici.
- Prikazan je deo ugrađenog koda u fajlu *switchCase.ino*.

while petlja

```
// while the button is pressed, take calibration readings:
while (digitalRead(buttonPin) == HIGH) {
  calibrate();
}
```

- Primer *while* petlje je prikazan na slici.
- Pokazan je ugrađen primer *WhileStatementConditional.iso*.
- Kakva se petlja dobija ako je uslov *while(1)*?
- Kakva se petlja dobija ako je uslov *while(303)*?

for petlja

```
const int kPinLed = 13;

void setup()
{
  pinMode(kPinLed, OUTPUT);
}

void loop()
{
  for(int i = 0; i < 4; i++){
    digitalWrite(kPinLed, HIGH);
    delay(200);
    digitalWrite(kPinLed, LOW);
    delay(200);
  }
  delay(1000); // 1 second
}
```

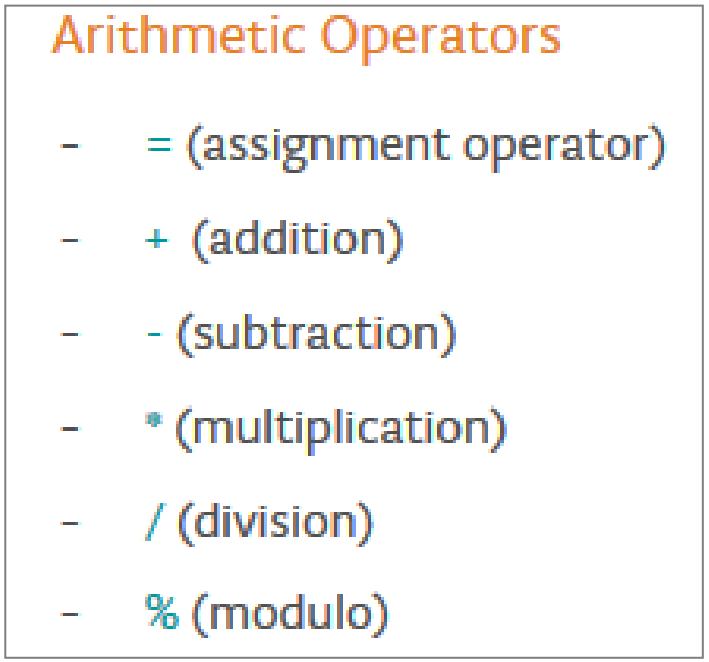
- Primer *for* petlje je prikazan na slici.
- Šta je rezultat koda sa slike?
- Kod je prezet iz knjige: Kod je pruzet iz knjige: Alan G. Smith, Introduction to Arduino: A piece of cake!, online: https://www.introtoarduino.com/downloads/IntroArduinoBook.pdf, 2011.
- Da li treba koristiti *i* kao iterator u petlji? Zašto?

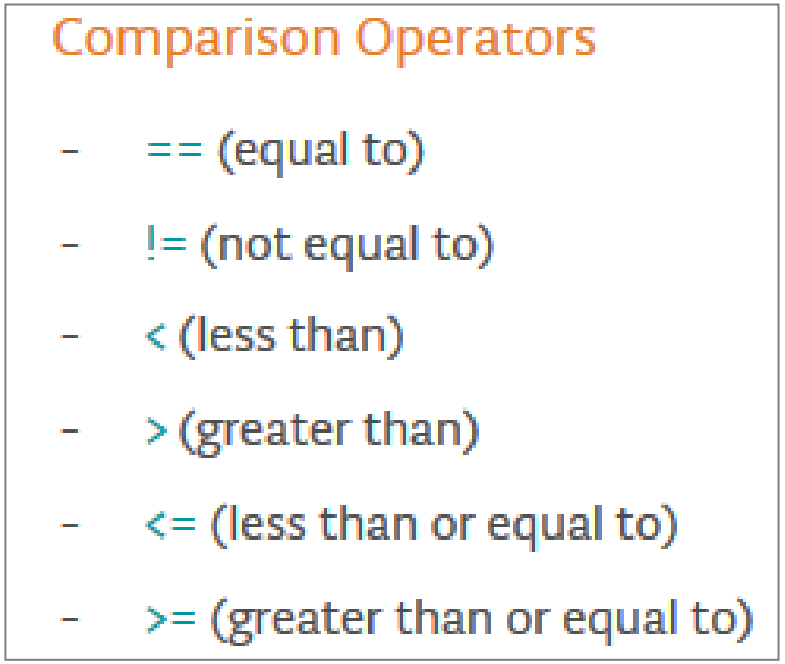
Šta je rezultat koda sa slike?

```
void loop()
{
    delayTime = delayTime - 100;
    if(delayTime <= 0){    // If the delay time is zero or ↩
        ↪ less, reset it.
      delayTime = 1000;
    }
    digitalWrite(kPinLed, HIGH);
    delay(delayTime);
    digitalWrite(kPinLed, LOW);
    delay(delayTime);
}
```

- Kod je pruzet iz knjige: Alan G. Smith, Introduction to Arduino: A piece of cake!, online: https://www.introtoarduino.com/downloads/IntroArduinoBook.pdf, 2011.
- Na slici nije prikazana *setup()* funkcija, ni definisanje globalnih promenljivih tj. prikazan je samo deo koda.

Aritmetički operatori i poređenja





- Operatori su prikazani na slikama.
- Više o operatorima na sajtu: https://www.arduino.cc/en/Reference/HomePage.
- Celobrojno deljenje se realizuje sa znakom ”/” ako su imenilac i brojilac celobrojne vrednosti.

Logički i složeni (eng. *compound*) operatori

Boolean Operators

- && (and)
- || (or)
- ! (not)

Bitwise Operators

- & (bitwise and)
- | (bitwise or)
- ^ (bitwise xor)
- ~ (bitwise not)
- << (bitshift left)
- >> (bitshift right)

Compound Operators

- ++ (increment)
- -- (decrement)
- += (compound addition)
- -= (compound subtraction)
- *= (compound multiplication)
- /= (compound division)
- %= (compound modulo)
- &= (compound bitwise and)
- |= (compound bitwise or)

- Operatori su prikazani na slikama.
- Više o operatorima na sajtu: https://www.arduino.cc/en/Reference/HomePage.

Tipovi podataka

Data Types

- void
- boolean
- char
- unsigned char
- byte
- int
- unsigned int
- word
- long
- unsigned long
- short
- float
- double
- string - char array
- String - object
- array

- Tipovi podataka su prikazani u tabeli.
- Detalji se mogu pogledati na: https://www.arduino.cc/en/Reference/HomePage.
- Konverzija se postiže primenom odgovarajućih funkcija.

Conversion

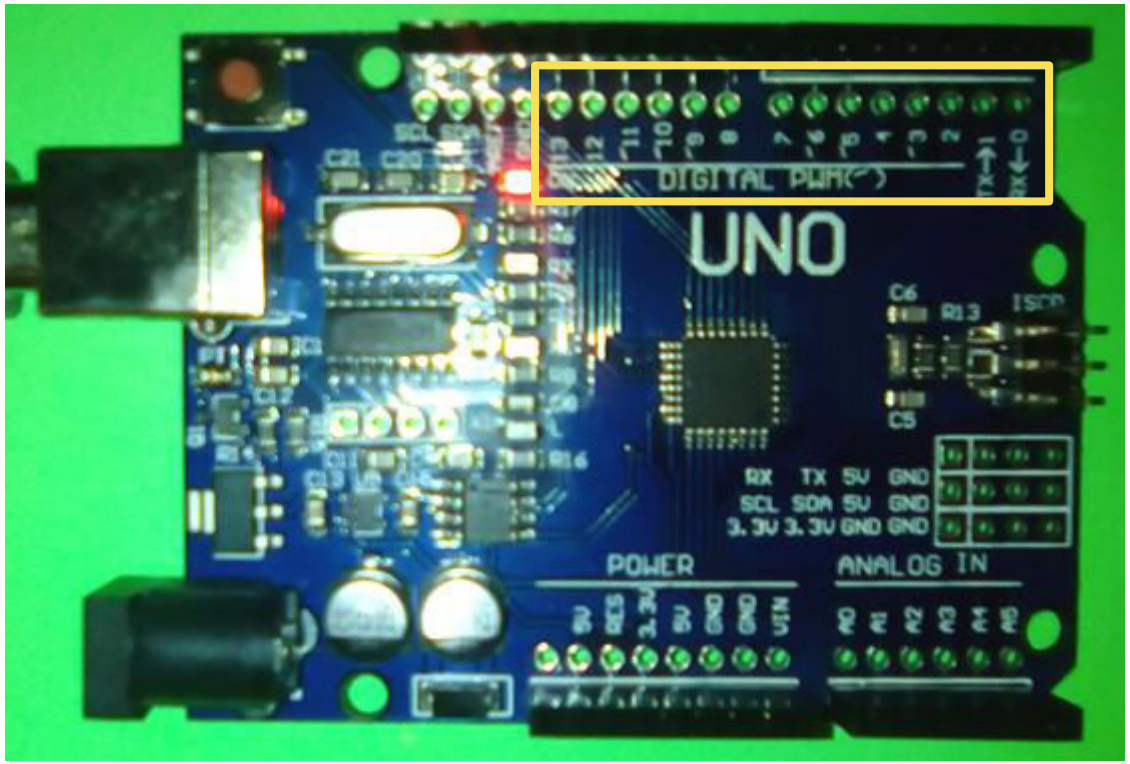
- char()
- byte()
- int()
- word()
- long()
- float()

Promena rezolucije

```
// change the resolution to 16 bits and read A0
analogReadResolution(16);
Serial.print(", 16-bit : ");
Serial.print(analogRead(A0));

// change the resolution to 8 bits and read A0
analogReadResolution(8);
Serial.print(", 8-bit : ");
Serial.println(analogRead(A0));
```

- Nije moguća na mikrokontrolerskim pločama koje se koriste na laboratorijskim vežbama (UNO R3 i sličnim).
- Međutim, moguća je na Arduino Due, Arduino Zero i drugim mikrokontrolerskim pločicama.
- Koristi se funkcija *analogReadResolution()*.
  - Više o ovoj funkciji na https://www.arduino.cc/en/Reference/AnalogReadResolution (slika je takođe preuzeta sa ovog sajta).
- U principu, kod neće javiti grešku ako se povećava rezolucija iznad hardverskih granica, ali će ti bitovi biti prazni.
  - Ovakav kod ima smisla pisati kada se planira povećanje hardverskog kapaciteta.

Pulse Width Modulation



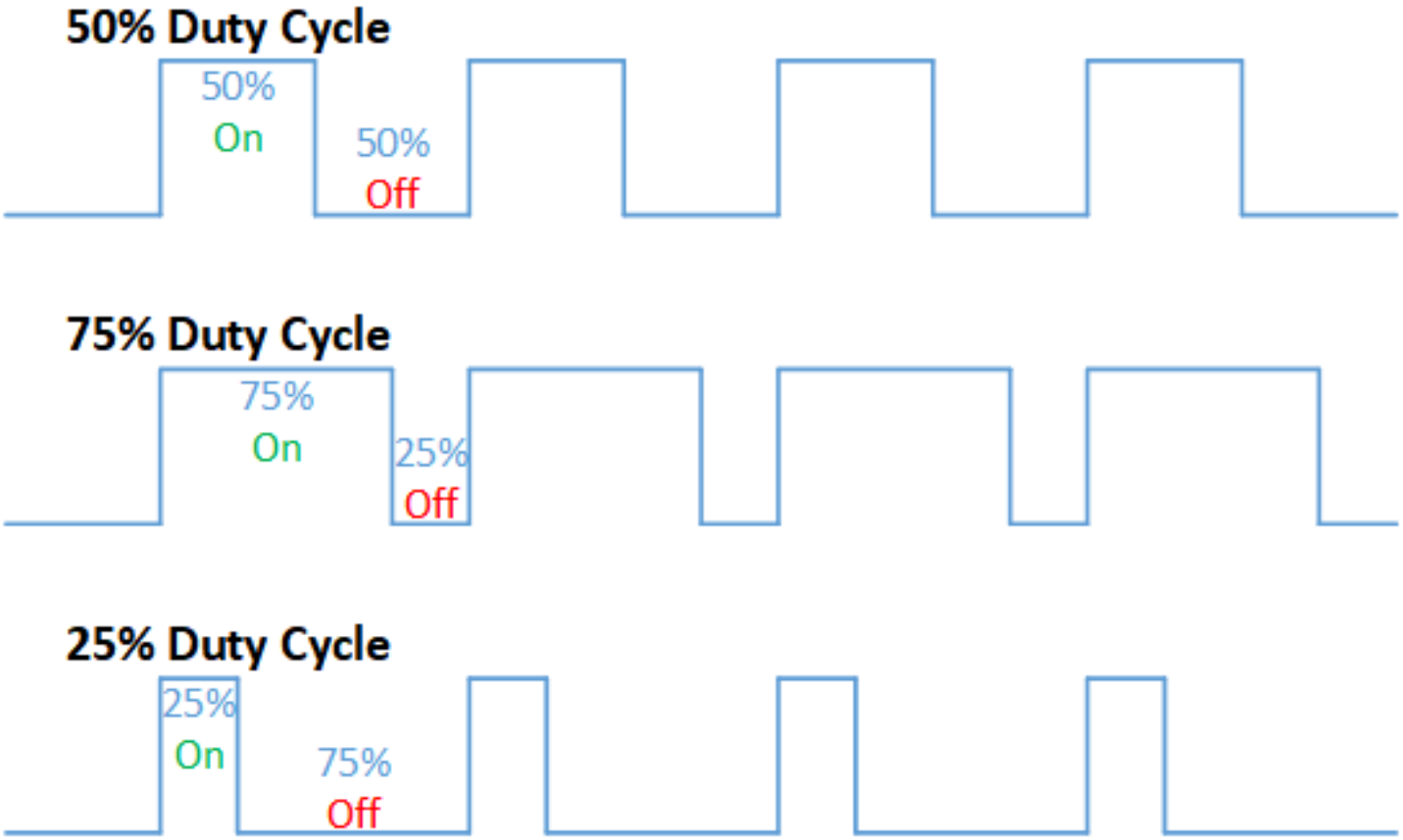

```
void loop() {
  digitalWrite(LED_BUILTIN, HIGH);
  delay(1000);
  digitalWrite(LED_BUILTIN, LOW);
  delay(1000);
}
```

- Umesto *digitalWrite()* koristi se *analogWrite()* funkcija. Funkcija *digitalWrite()* je korišćena kod *Blink.iso* primera.
- Na pločici piše pored digitalnih pinova PWM (pogledati sliku).
- To znači mogućnost da se upravlja sa *duty cycle*.

PWM definicija/e

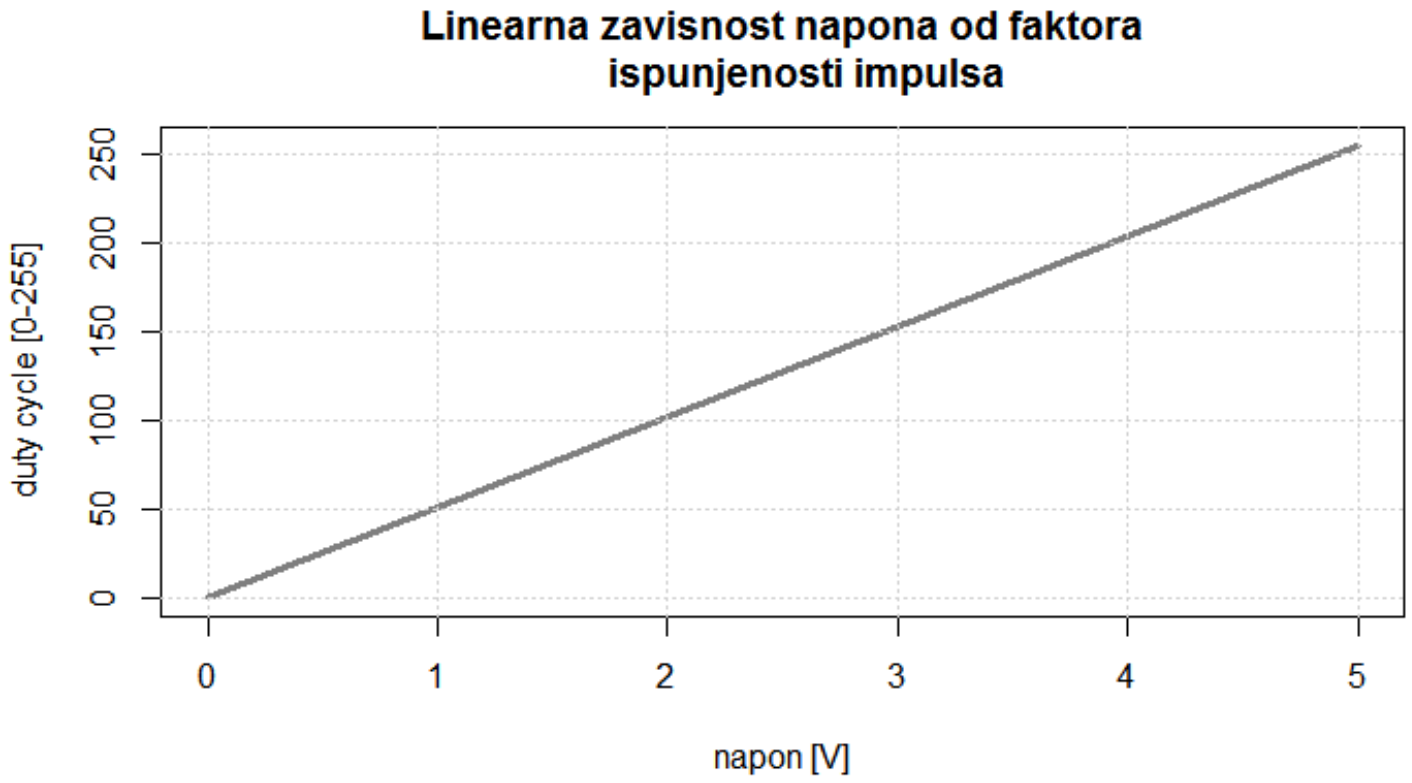
- “Pulse Width Modulation (PWM, http://en.wikipedia.org/wiki/Pulse-width_modulation) is a fancy term for describing a type of digital signal. Pulse width modulation is used in a variety of applications including sophisticated control circuitry” (izvor: https://learn.sparkfun.com/tutorials/pulse-width-modulation).
- “Pulse-width modulation (PWM), or pulse-duration modulation (PDM), is a modulation technique used to encode a message into a pulsing signal” (izvor: https://en.wikipedia.org/wiki/Pulse-width_modulation).
- “Pulse Width Modulation, or PWM, is a technique for getting analog results with digital means. Digital control is used to create a square wave, a signal switched between on and off. This on-off pattern can simulate voltages in between full on (5 Volts) and off (0 Volts) by changing the portion of the time the signal spends on versus the time that the signal spends off” (izvor: https://www.arduino.cc/en/Tutorial/PWM).

PWM u praksi



By Thewrightstuff - Own work, CC BY-SA 4.0, https://commons.wikimedia.org/w/index.php?curid=72876123

PWM vrednosti 0-255 na 0-5 V



- Linearna skala je prikazana na slici.
- Skala je nacrtana u R-u (https://www.r-project.org/about.html).
- Opseg od 0 do 255 se odnosi na *duty cycle* tj. faktor ispunjenosti impulsa.

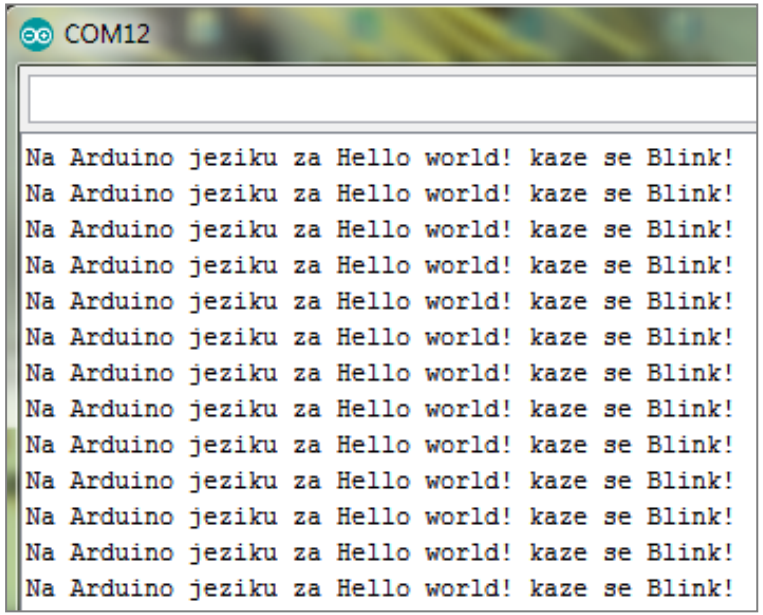
analogWrite() primer

```
int led = 9;           // the PWM pin the LED is attached to
int brightness = 0;    // how bright the LED is
int fadeAmount = 5;    // how many points to fade the LED by

// the setup routine runs once when you press reset:
void setup() {
  // declare pin 9 to be an output:
  pinMode(led, OUTPUT);
}

// the loop routine runs over and over again forever:
void loop() {
  // set the brightness of pin 9:
  analogWrite(led, brightness);

  // change the brightness for next time through the loop:
  brightness = brightness + fadeAmount;

  // reverse the direction of the fading at the ends of the fade:
  if (brightness <= 0 || brightness >= 255) {
    fadeAmount = -fadeAmount;
  }
  // wait for 30 milliseconds to see the dimming effect
  delay(30);
}
```

- Ugrađeni primer *Fade.iso* je prikazan na sclici.
- Šta je funkcija ovog primera?
- Kada se koristi *analogWrite()* funkcija, nije potrebno koristiti *pinMode()* funkciju.
- Koristi se za kontrolu osvetljenosti (intenziteta svetlosti na diodi), ali i za kontrolu motora.

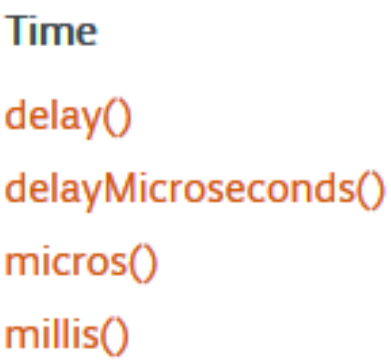
Primer poruke na serijskom portu

```
void setup() {
  Serial.begin(9600);
}

void loop() {
  Serial.println("Na Arduino jeziku za Hello world! kaze se Blink!");
  delay(100);
}
```



- Na slici je prikazan primer ispisa poruke na serijskom portu.
- Dve osnovne funkcije za rad sa stringovima su *Serial.print()* i *Serial.println()*.
- Šta bi bilo da je iskorišćena funkcija *Serial.print()*?

Tajmer



- Funkcije koje se koriste za merenje vremena su prikazane na slici (https://www.arduino.cc/reference/en/).
- Pogledati na slici ispod upozorenje kod korišćenja *delay()* funkcije (https://www.arduino.cc/reference/en/language/functions/time/delay/).

Notes and Warnings

While it is easy to create a blinking LED with the `delay()` function, and many sketches use short delays for such tasks as switch debouncing, the use of `delay()` in a sketch has significant drawbacks. No other reading of sensors, mathematical calculations, or pin manipulation can go on during the delay function, so in effect, it brings most other activity to a halt. For alternative approaches to controlling timing see the millis() function and the sketch sited below. More knowledgeable programmers usually avoid the use of `delay()` for timing of events longer than 10's of milliseconds unless the Arduino sketch is very simple.

Certain things do go on while the delay() function is controlling the Atmega chip however, because the delay function does not disable interrupts. Serial communication that appears at the RX pin is recorded, PWM (analogWrite) values and pin states are maintained, and interrupts will work as they should.

millis() i *micros()* funkcije

micros()

[Time]

Description

Returns the number of microseconds since the Arduino board began running the current program. This number will overflow (go back to zero), after approximately 70 minutes. On 16 MHz Arduino boards (e.g. Duemilanove and Nano), this function has a resolution of four microseconds (i.e. the value returned is always a multiple of four). On 8 MHz Arduino boards (e.g. the LilyPad), this function has a resolution of eight microseconds.

millis()

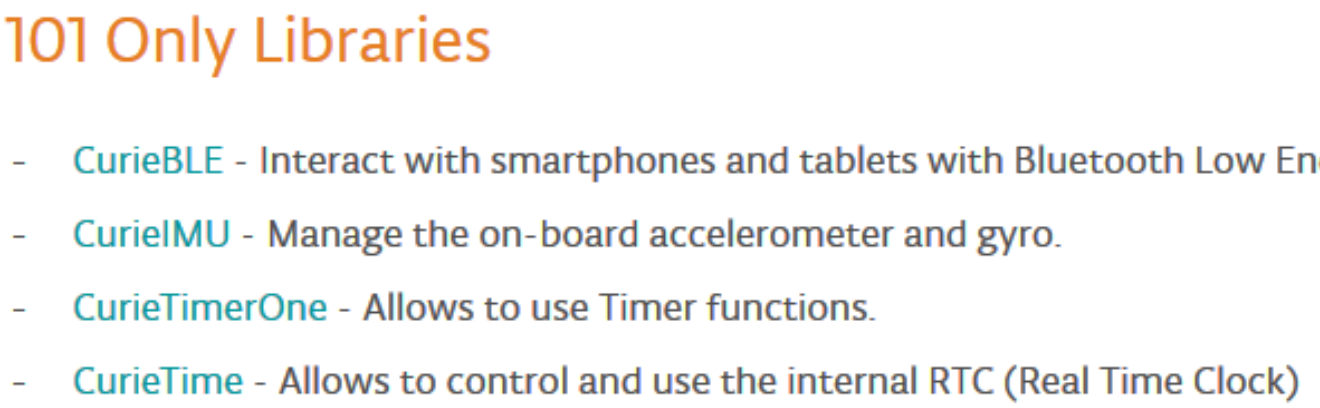
[Time]

Description

Returns the number of milliseconds since the Arduino board began running the current program. This number will overflow (go back to zero), after approximately 50 days.

- Za UNO R3 (Mega328P) *clock* frekvencija je 250 kHz. Dodatno, 32 b je rezervisano za *timer*.
- Pogledati detaljno uputstvo za ove dve funkcije pre primene (https://www.arduino.cc/reference/en/language/functions/time/millis/, https://www.arduino.cc/reference/en/language/functions/time/micros/).

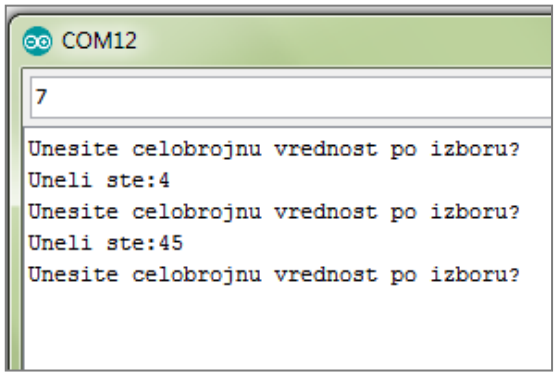
Druge Arduino biblioteke



- Arduino okruženje može biti "proširenо" korišćenjem biblioteka.
- Kako bi se uvezla neka biblioteka koristi se padajući meni *Sketch* i opcija *Import Library*.
- Za one koji to žele, postoje i uputstva kako napisati biblioteku za Arduino: https://www.arduino.cc/en/Hacking/LibraryTutorial.
- Primeri nekih biblioteka su prikazani na slici (https://www.arduino.cc/en/Reference/Libraries).
- Jedan savet: Nikada nemojte u potpunosti "verovati" nekoj biblioteci, proverite šta tačno radi koja funkcija, pre nego što je iskoristite. Nekada brza rešenja nisu najbolja rešenja.

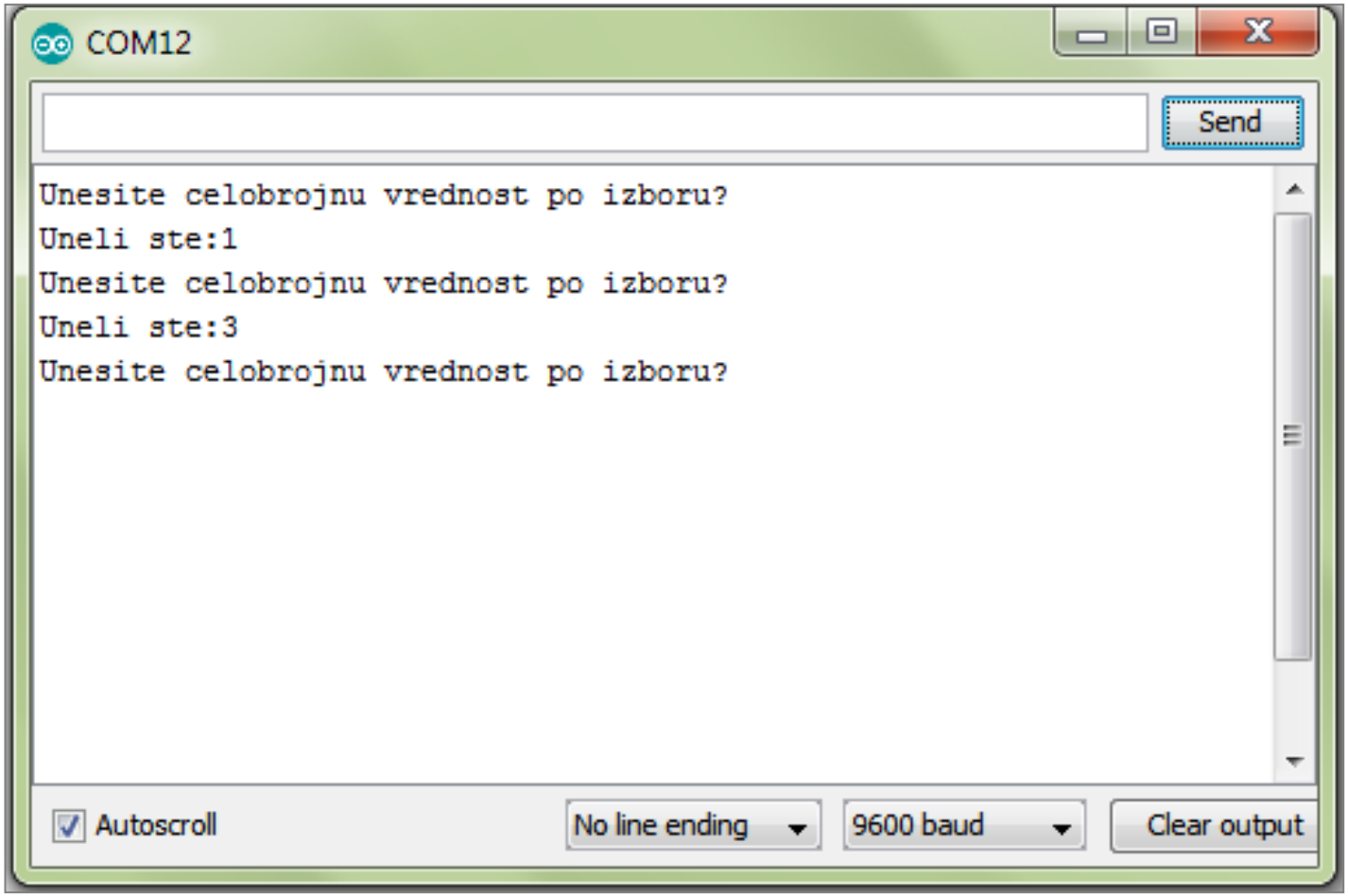
Korisnički unos u Arduino programu

```
/*
  Primer korisničkog unosa parametara.
*/
int broj;  // celobrojna vrednost po izboru korisnika

void setup() {
  // serijski port se koristi za unos broja
  Serial.begin(9600);
}

void loop() {
  // stampanje pitanja na serijskom portu (Serial monitor)
  Serial.println("Unesite celobrojnu vrednost po izboru?");
  while(Serial.available() == 0) { // ceka se unos korisnika
  }
  broj = Serial.parseInt();  // ocitava se vrednost koju unosi korisnik
  // prikazuje se sta je uneo korisnik na monitoru
  Serial.print("Uneli ste: ");
  Serial.println(broj);
}
```



- Moguće je realizovati takav program da korisnik unosi parametre preko serijskog porta.
- Koristi se funkcija *Serial.parseInt()* za koju uputstvo možete pogledati na: https://www.arduino.cc/en/Serial/ParseInt.
- Primer koda i izgled serijskog monitora dati su na slici.

Unos 1.3?


COM12
Send
Unesite celobrojnu vrednost po izboru?
Uneli ste:1
Unesite celobrojnu vrednost po izboru?
Uneli ste:3
Unesite celobrojnu vrednost po izboru?
Autoscroll
No line ending
9600 baud
Clear output

PAJTON PROGRAMIRANJE

Pajton

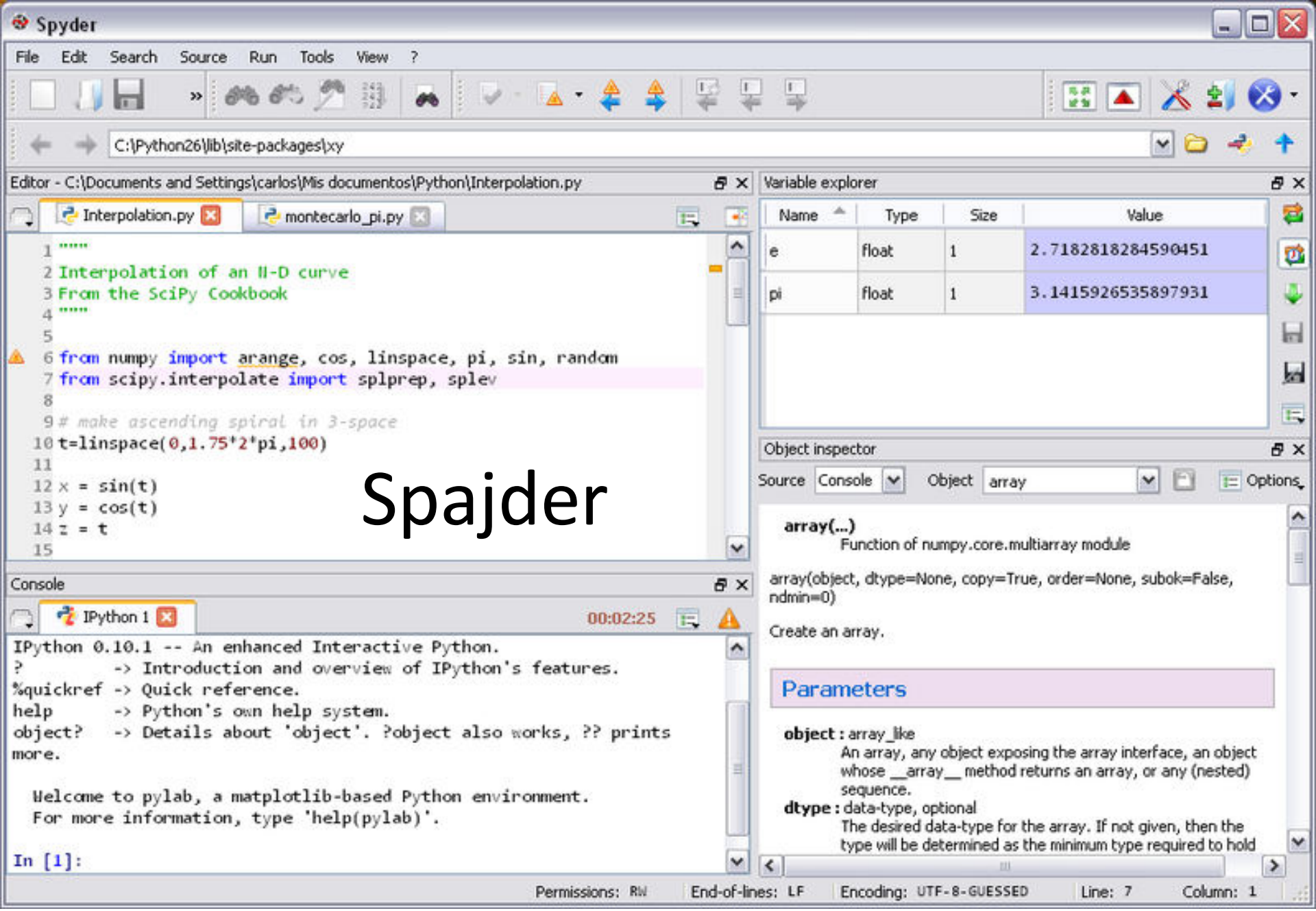
- Pajton je programski jezik (https://en.wikipedia.org/wiki/Python_(programming_language)). Slobodan softver (= *open source*). Modularan. Praktičan. Ovde: za merenja sa serijskog porta, vizuelizaciju podataka i dr.
- Na KLIN predmetu su instalirane jedna po jedna biblioteka. Sada, pomoću pip-a možete instalirati koje god želite i/ili koje su Vam potrebne.
- Međutim, postoje i tzv. *Custom Distributions* instalacije koje dolaze sa skupom paketa, npr.:
  - Python(x,y), http://www.pythonxy.com/, dec. 2017.
  - Sage, http://www.sagemath.org/, dec. 2017.
- Više programskih paradigmi je "pomešano" u ovom programskom jeziku: objektno-orijentisano, funkcionalno programiranje, ...
- **VAŽNO: Ova prezentacija nema zadatak da uči studentkinje i studente programiranju, već da istakne neke elemente programskog jezika Pajton s obzirom da je to sve popularniji programski jezik u merenjima koja su zasnovana na primeni računara.**

Pajton – deo istorije

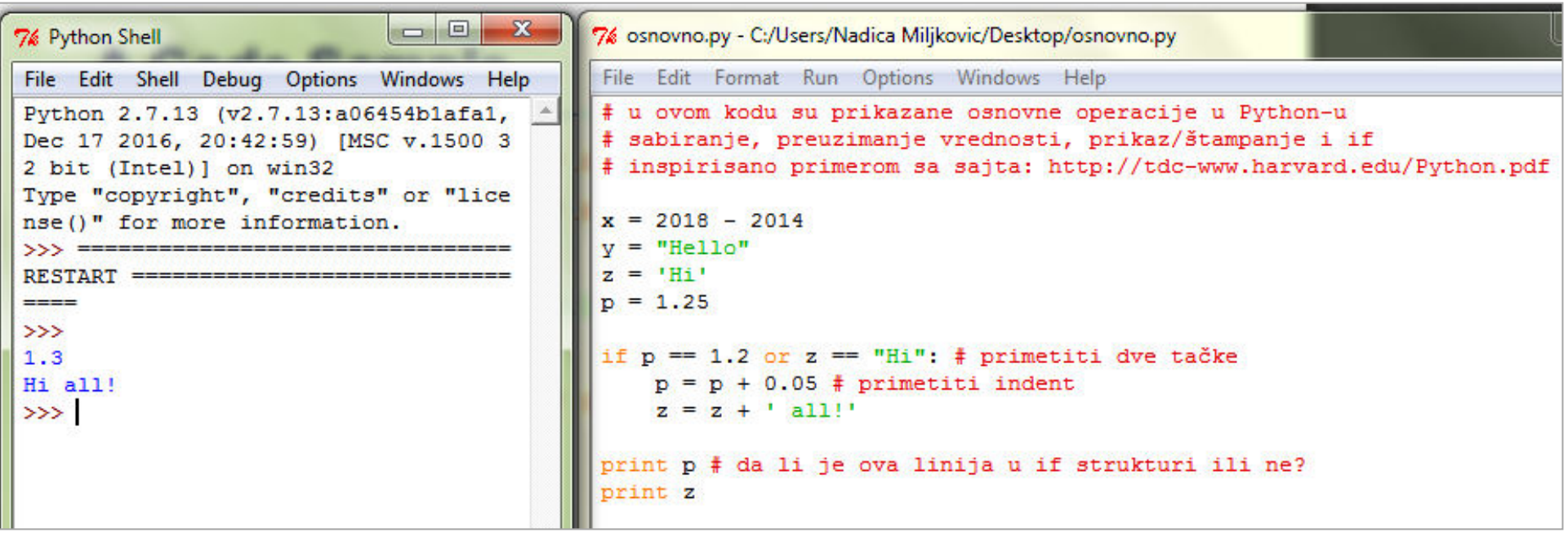
- Kreirano ga je Guido van Rossum i prva verzija je objavljena 1991. godine.
- Guido je rekao o kreiranju Pajtona (https://en.wikipedia.org/wiki/History_of_Python):

> ...In December 1989, I was looking for a "hobby" programming project that would keep me occupied during the week around Christmas. My office ... would be closed, but I had a home computer, and not much else on my hands. I decided to write an interpreter for the new scripting language I had been thinking about lately: a descendant of ABC that would appeal to Unix/C hackers. I chose Python as a working title for the project, being in a slightly irreverent mood (and a big fan of *Monty Python's Flying Circus*).

- Realno? Svima želim *Merry Christmas*!



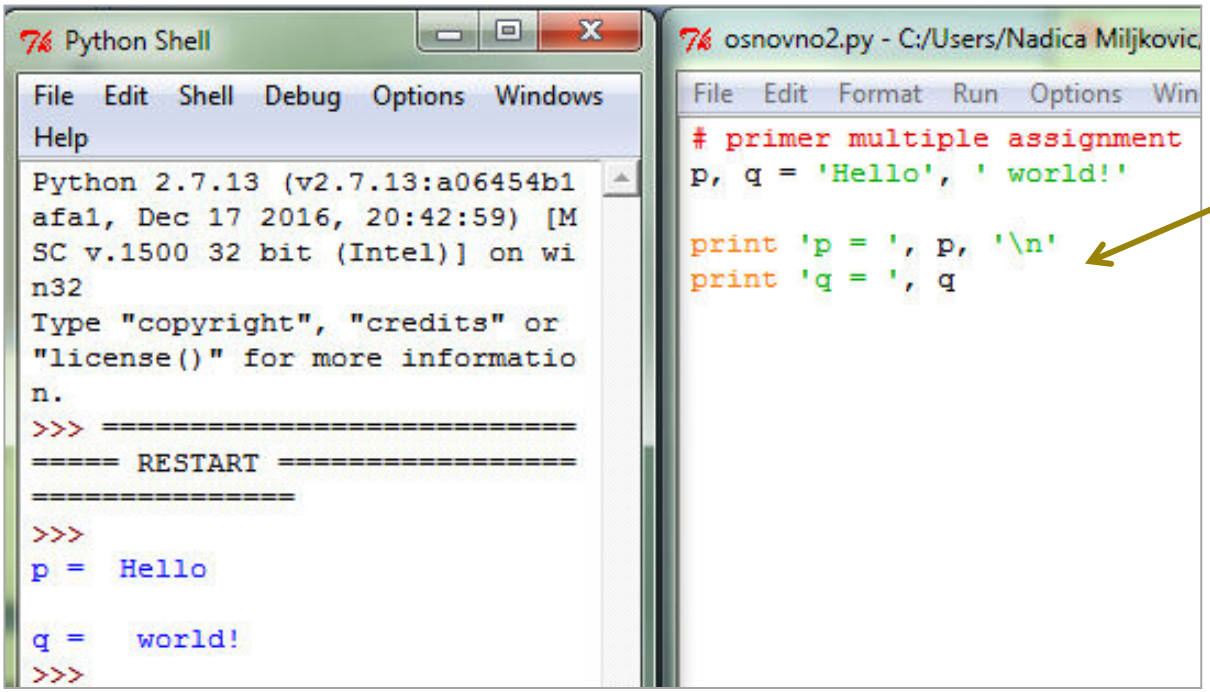
Pajton osnove



- Primetiti da: na kraju izraza nema “;”, da je string moguće definisati na dva načina, kako se izgleda *if* struktura, da ne postoje posebne komande za početak i kraj, kako izgledaju logički operatori?
- Dodatno, primetiti da se za spajanje stringova (eng. *concatenation*) koristi operator “+”. Negaciji odgovara operator *not*.
- VAŽNO: kod prve dodele vrednosti nekoj promenljivoj dolazi do njene inicijalizacije.
- Na KLIN predmetu se koristi klasičan IDE za Pajton (eng. *Integrated Development Environment*). Prikazani su primeri za Pajton 2.

Pajton, osnovni tipovi podataka

```
""" Ovo je primer višelinijskoj komentara
koji se može koristiti po želji"""
print y
# jednolinijski komentar
```



prelazak u novu liniju i više parametara za prikaz...

- Celobrojni tip podataka (eng. *integer*) je podrazumevani tip za sve brojeve.
- Za *string*-ove se mogu koristiti “”, ali i ‘’.
- Jednolinijski komentari počinju sa tarabom “#”. Međutim, višelinijski komentari počinju sa “”” (pogledati sliku gore).
- *Multiple assignment* je jedna od osobina koda u Pajtonu (pogledati sliku dole).

Liste: *append* i *insert*

```
# append i insert funkcije
print 'x = ', x

x.append(56)
print 'x.append(56) = ', x

x.insert(1, 55)
print 'x.insert(1, 55) = ', x
```

```
x =  [10, 9, 8, 7, 6]
x.append(56) =  [10, 9, 8, 7, 6, 56]
x.insert(1, 55) =  [10, 55, 9, 8, 7, 6, 56]
>>>
```

- Primeri primene funkcija *append* i *insert* nad listama su prikazani.
- Primetiti da se pozivaju sa tačkom nakon promenljive i da se nova vrednost dodeljuje listi za koju se poziva.
- Dodatno postoje i: *index* (na izlazu daje indeks na kome se prvi put pojavljuje element koji je prosleđen kao argument), *count* (broj pojavljivanja arguementa), *remove* (briše se prvo pojavljivanje argumenta), *reverse* (menja se redosled elemenata), *sort*, *clear*, *copy*, ...
- Za konverziju npr. u listu koristi se istoimena funkcija: *list()*.

Kontrola toka i strukture u Pajtonu

```
sensor = 1;
if sensor > 2:
    print 'Upalite svetlo!'
elif sensor < 0.5:
    print 'Ugasite svetlo!'
else:
    print 'Neka ostane kako je bilo'
print 'Ovo je izvan if strukture i štampa se sigurno'

vr = 1
while vr < 5:
    if vr >= 1:
        vr += 0.5
        continue
    vr = vr + 1
    if vr > 4.5:
        break

for x in range(15):
    if x < 3:
        x += 1
        continue
    print(x)
```

```
Neka ostane kako je bilo
Ovo je izvan if strukture i štampa se sigurno
3
4
5
6
7
8
9
10
11
12
13
14
>>>
```

- Na slici su prikazane osnovne strukture u Pajtonu.
- Obratiti pažnju na ":" na kraju *if*, *while* i *for* strukture!
- Šta je rezultat koda sa slike? Koliko je *vr* na kraju *while* petlje?

Klase i objekti

```
class Oscilloscope(object):
    #
    def __init__(self, address = '/dev/ttyS0'):
        try:
            self.port = serial.Serial(address)
            self.port.timeout = 5
            scopeid = self.ask('*idn?')
            if scopeid == '':
                print 'Cannot find oscilloscope at', address
            self.write('header 0')
            self.write('data:encdg ascii')
        except:
            print
            print 'Cannot open port', address
            print
    #
    def __del__(self):
        self.port.close()
    #
    def write(self, string):
        self.port.write(string + '\n')
    #
    def read(self):
        return self.port.readline()
    #
    def readchar(self):
        return self.port.read()
    #
    def ask(self, question):
        self.write(question)
        return self.read()
```

- Enkapsulacija, polimorfizam, nasleđivanje ... poznato?
- Na slici je dat primer definisanja klase osciloskop u kodu za predmet Električna merenja na Katedri za elektroniku: http://tnt.etf.rs/~oe2em/.

Neke korisne funkcije

- *len()*
- repeat()
- *ceil()*
- *radians()*
- *sin()*
- *exp()*
- Konstante: *pi* i *e*
- *capitalize()*
- *islower()*
- *ilsupper()*
- *split()*
- *read()*
- *write()*
- *close()*
- Preporučene liste funkcija:
  - http://www.astro.up.pt/~sousasag/Python_For_Astronomers/Python_qr.pdf
  - https://perso.limsi.fr/pointal/_media/python:cours:abregepython-english.pdf

```
# Python 3: Fibonacci series up to n
>>> def fib(n):
>>>     a, b = 0, 1
>>>     while a < n:
>>>         print(a, end=' ')
>>>         a, b = b, a+b
>>>     print()
>>> fib(1000)
0 1 1 2 3 5 8 13 21 34 55 89 144 233 377 610 987
```

```
# Python 3: Simple arithmetic
>>> 1 / 2
0.5
>>> 2 ** 3
8
>>> 17 / 3  # classic division returns a float
5.666666666666667
>>> 17 // 3  # floor division
5
```

```
# Python 3: List comprehensions
>>> fruits = ['Banana', 'Apple', 'Lime']
>>> loud_fruits = [fruit.upper() for fruit in
fruits]
>>> print(loud_fruits)
['BANANA', 'APPLE', 'LIME']

# List and the enumerate function
>>> list(enumerate(fruits))
[(0, 'Banana'), (1, 'Apple'), (2, 'Lime')]
```

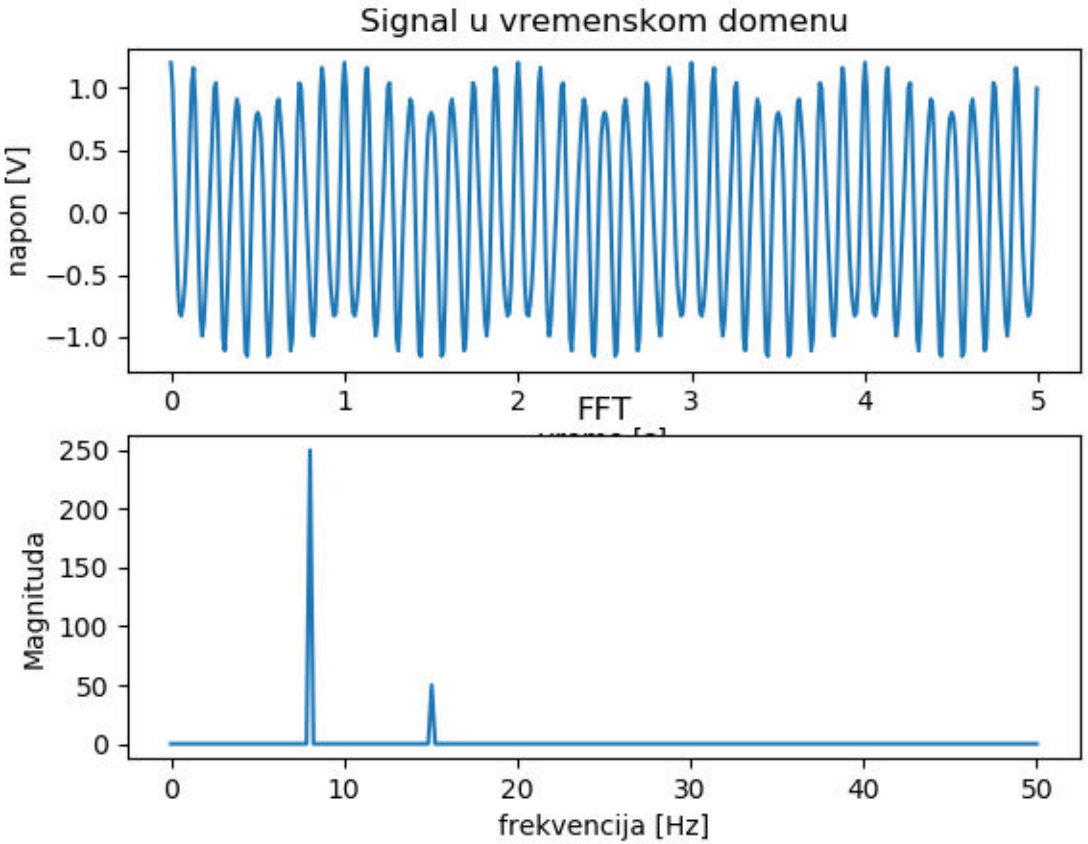
```
# Python 3: Simple output (with Unicode)
>>> print("Hello, I'm Python!")
Hello, I'm Python!

# Input, assignment
>>> name = input('What is your name?\n')
>>> print('Hi, %s.' % name)
What is your name?
Python
Hi, Python.
```

```
# For loop on a list
>>> numbers = [2, 4, 6, 8]
>>> product = 1
>>> for number in numbers:
...     product = product * number
...
>>> print('The product is:', product)
The product is: 384
```

Online console from PythonAnywhere

https://www.python.org/

numpy



- Kaže se da ova biblioteka sadrži funkcije koje omogućavaju funkcionalnost sličnu Matlabu (relativno brze operacije sa nizovima i matricama, linearna algebra)
- Više informacija na: http://www.numpy.org/
- Primer računanja FFT (eng. *Fast Fourier Transform*) za jedan sintetički signal je prikazan na slici.

```
# ovaj kod je inspirisan online primerom sa sajta:
# https://stackoverflow.com/questions/15382076/plotting-power-spectrum-in-python

import numpy as np
import matplotlib.pyplot as plt

fs = 100.0 # definisati frekvenciju odabiranja
t = np.arange(0, 5, 1/fs) # kreirati vreme u trajanju od 5 sekundi
# zbor kosinusa sa osnovnim frekvencijama od 8 i 15 Hz
x = np.cos(2 * np.pi * 8 * t) + 0.2 * np.cos(2 * np.pi * 15 * t)

# Furijeova transformacija
xF = np.fft.fft(x)
N = len(xF)
xF = xF[0:N/2] # posmatra se do Nikvista (fs/2)
fr = np.linspace(0, fs/2, N/2) # definiše se frekvencijska karakteristika

# crtanje grafika
fig = plt.figure()

ax = fig.add_subplot(211)
ax.plot(t, x)
plt.title('Signal u vremenskom domenu')
plt.xlabel('vreme [s]')
plt.ylabel('napon [V]')

ax = fig.add_subplot(212)
ax.plot(fr, abs(xF))
plt.title('FFT')
plt.ylabel('Magnituda')
plt.xlabel('frekvencija [Hz]')

plt.show()
```

A kod?

pyserial

- Više na: https://pythonhosted.org/pyserial/.
- Ovo je biblioteka koja omogućava pristup serijskom portu.
- Moguće je čitati podatke (eng. *read*) sa serijskog porta, ali ih je moguće i "slati" tj. upisivati na serijski port (eng. *write*).
- Za one koje zanima da pristupaju virtuelnim serijskim portovima, mogu pogledati: http://com0com.sourceforge.net/.
- Način na koji mi pristupamo podacima iz Pajtona preko serijskog porta se naziva snifovanje (eng. *sniffing*).
- To se koristi za merenja u KLIN!

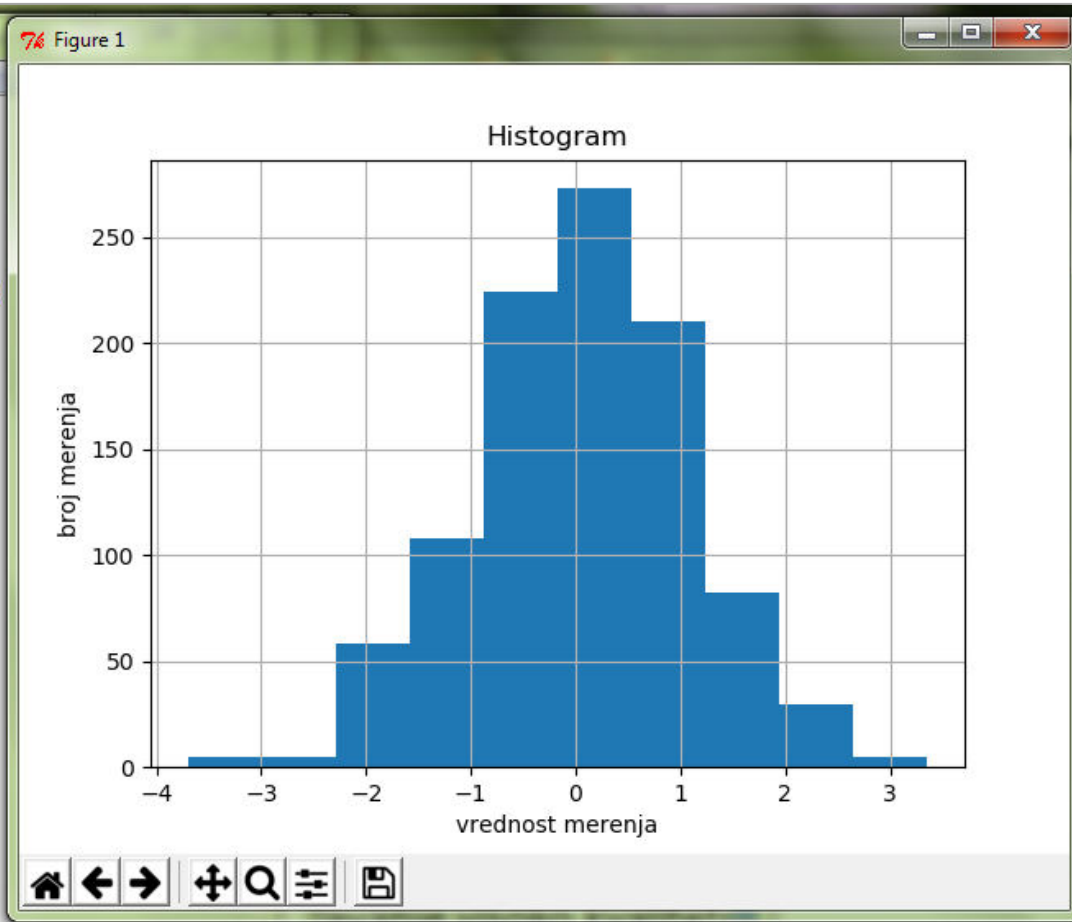
matplotlib

- Ova biblioteka sadrži niz funkcija koje omogućavaju iscrtavanje visoko kvalitetnih grafika.
- Osim prikaza 2D grafika, koristi se i za prikaz 3D grafika, ali i za prikaz slika.
- Ako zatreba, horizontalni barovi se prikazuju primenom *barh()* funkcije.
- Više na: http://matplotlib.org/.

racun2.py - C:\Users\Nadica Miljkovic\Desktop\MSR2017\skripta\racun2.py

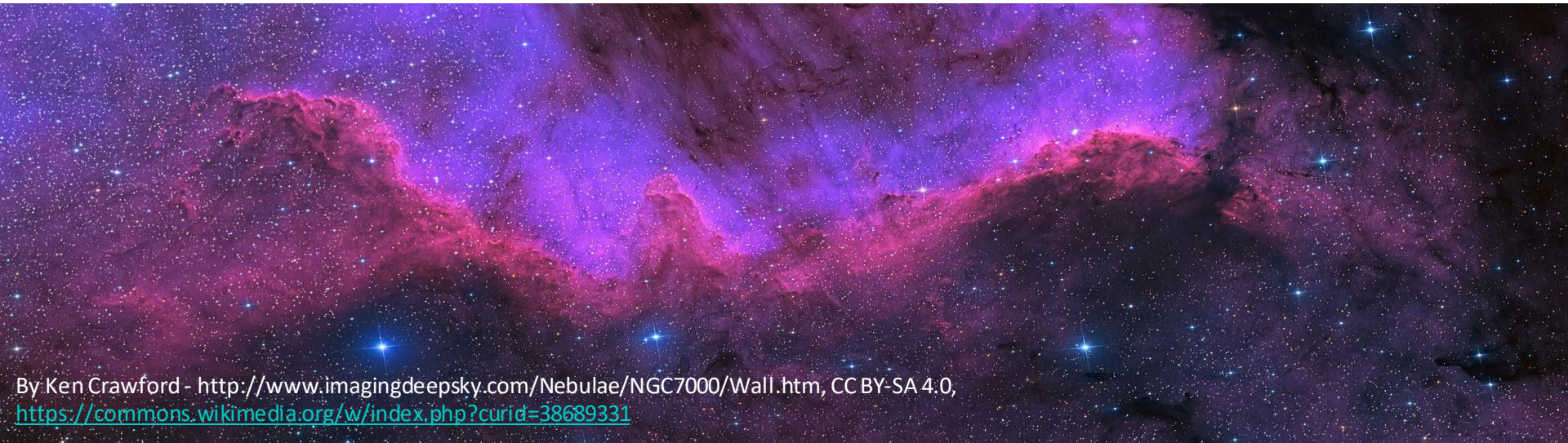
File Edit Format Run Options Windows Help

```
import numpy as np
import matplotlib.pyplot as plt

# promenljiva u koju se smeštaju učitane vrednosti
mer = []

# čitanje podataka iz fajla
f = open('C:/Users/Nadica Miljkovic/Desktop/merenje2.txt', 'r')
for line in f:
    mer.append(float(line.strip()))

# korišćenje numpy biblioteke za osnovne parametre
print np.mean(mer)
print np.std(mer)

# prikaz histograma
plt.hist(mer)
plt.xlabel('vrednost merenja')
plt.ylabel('broj merenja')
plt.grid(True)
plt.title('Histogram')
plt.show()
```



Za pripremu ove prezentacije

- korišćeni su materijali sa sledećih sajtova:
  - Merni sistemi u računarstvu, 13E053MSR, http://automatika.etf.rs/sr/13e053msr,
  - Introduction to Python, http://tdc-www.harvard.edu/Python.pdf, pristupljeno decembra 2017
  - http://matplotlib.org/tutorials/introductory/sample_plots.html#sphx-glr-tutorials-introductory-sample-plots-py, pristupljeno decembra 2017
  - i drugih koji su izlistani u prethodnim slajdovima.



ZADACI ZA RAD

Zadaci, ARDUINO

Ja ću Vam pokazati:

1. Pokrenuti ugrađenu funkciju “Blink.ino”. Testirati rad ove funkcije za različite parametre.
2. Pokrenuti ugrađenu funkciju “AnalogReadSerial.ino”. Testirati njen rad, a potom uneti izmenu tako da se može koristiti za prikaz napona na serijskom portu.
3. Testirati rad ugrađenog primera “Fade.ino”.
4. Kako se instalira biblioteka u Pajtonu, a kako u Arduinu?

Vi ćete testirati kada dođete u laboratoriju:

1. Na analogni ulaz povezati potenciometar. Dodatno, ako je vrednost veća od nekog unapred definisanog praga, uključiti ugrađenu diodu.
2. Testirati korisnički unos za paljenje/gašenje LED.
3. Kako bi se omogućilo merenje u Pajtonu?