Inspectoratul Școlar Județean Argeș

CONCURS DE ROBOTICĂ

RoboSmart

Ediția 1

2017-2018

Universitatea din Pitești

Facultatea de Electronică, Comunicații și Calculatoare

Web info aici:

Sponsori

Continental

TEXTRON FLEET MANAGEMENT

CURSUL 4
PROGRAMARE ARDUINO

- Să recapitulăm:
  - Resursele unui microsistem și asocierea lor cu programarea
  - Arduino IDE
  - Scheletul unui program Arduino
- Diferențe compilatoare
- Variabile. Tipuri de date. Constante
- Vectori
- Utilizarea librăriilor pentru accesarea circuitelor periferice
- Instrucțiuni:
  - Condiționare (if, switch)
  - Bucle (for, while)
  - Construirea și utilizarea funcțiilor
  - Structuri de date
- Lucru cu întreruperi

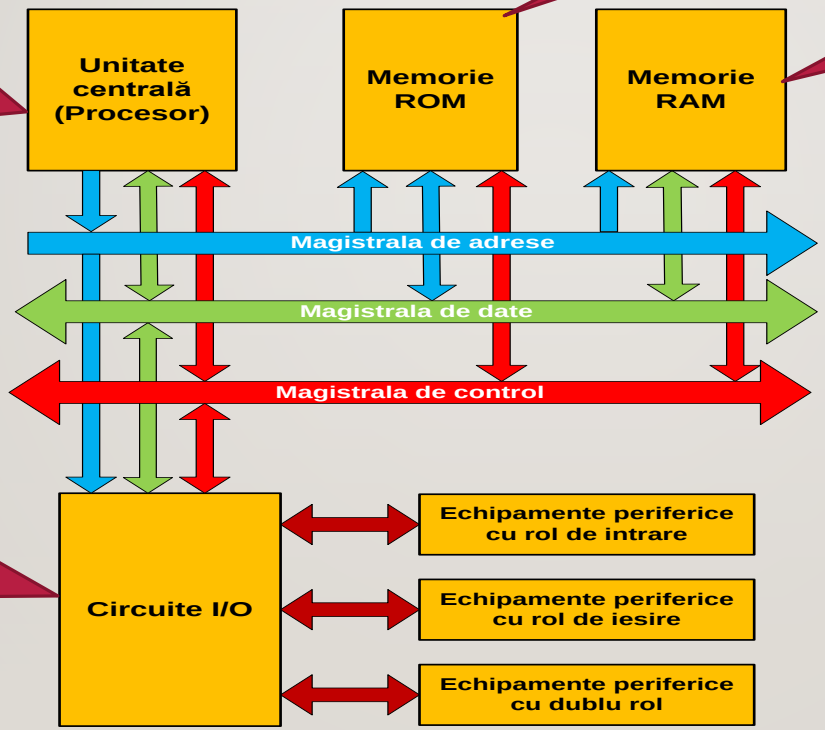
SĂ RECAPITULĂM...

- Resursele unui microsistem

Programul (cod mașină) + constante

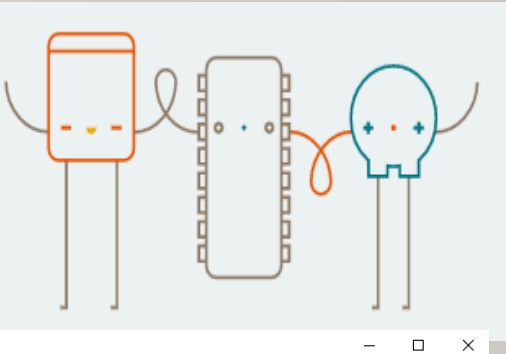
Variabilele

Aici se execută programul



Interfețe: USB, digital I/O, analogic...

https://www.arduino.cc/en/Main/Donate



SĂ RECAPITULĂM...

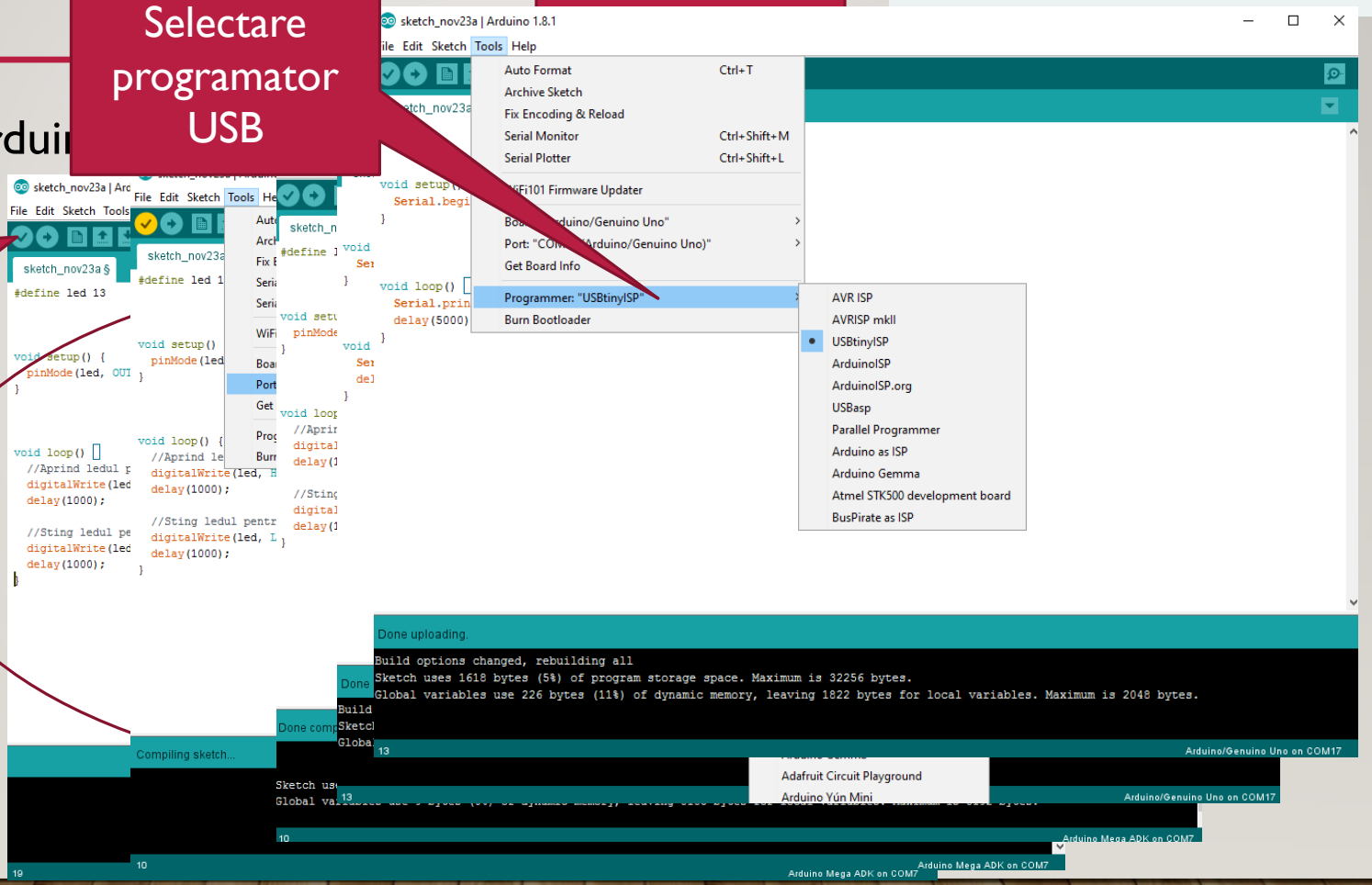
- Arduin

Deschiderea

Selectare programator USB

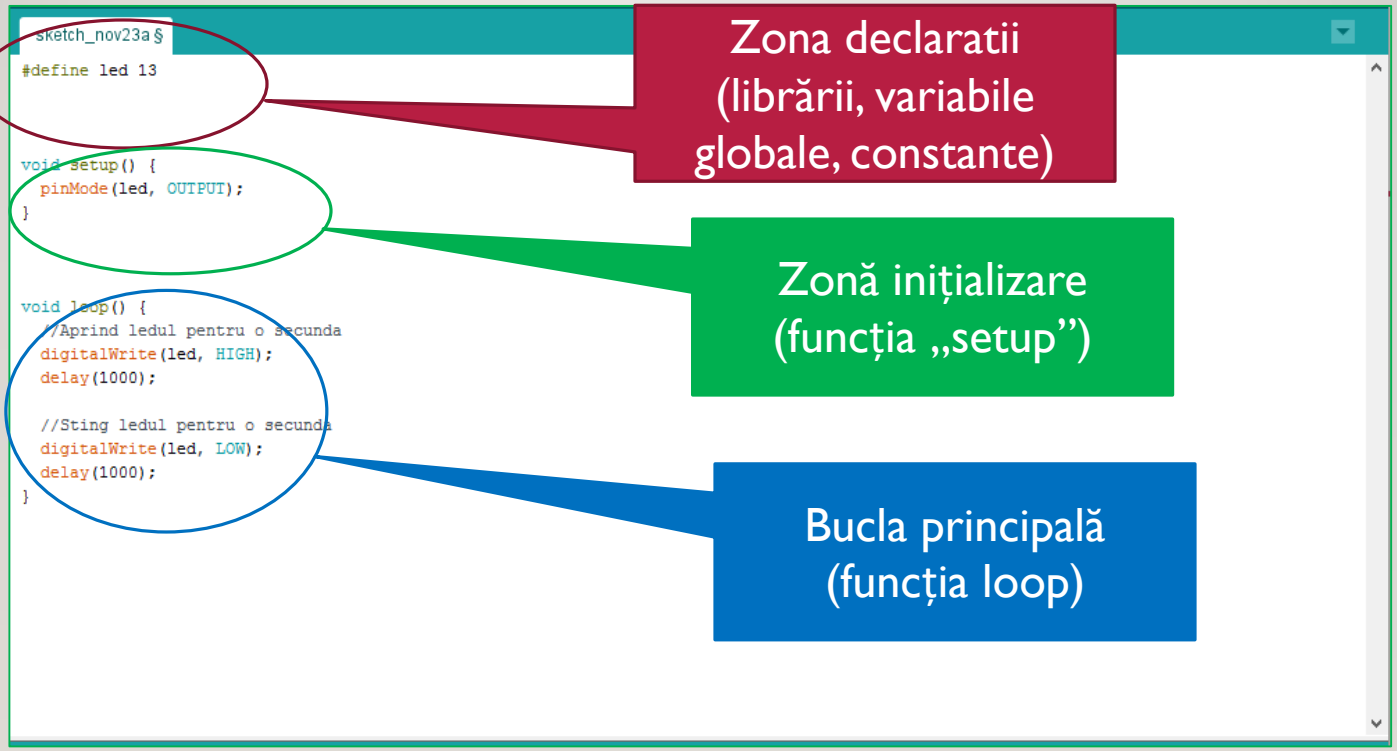
Bara cu instrumente (Verify, Upload)

Aici se scrie programul



SĂ RECAPITULĂM...

- Scheletul unui program Arduino



DIFERENȚE COMPILATOARE (ARDUINO VS. CODEBLOCKS ...)

- Câteva diferențe notabile:
  - Unde este consola de intrare/ieșire??
  - Funcția loop() vs. main(). Funcția setup.
  - Dimensiunea tipurilor de date. La Arduino:
    - char, byte: 8 biți;
    - boolean: 1 bit;
    - int, unsigned int: 16 biți;
    - long: 32 biți;
    - Float: 32 biți;
    - Double: 32 biți.
  - Amplasarea prototipului funcției.
  - Lucrul cu întreruperi.
  - Nu mai este necesară declarația „volatile” la variabilele folosite în subrutine întreruperi

UTILIZAREA LIBRĂRIILOR PENTRU ACCESAREA CIRCUITELOR PERIFERICE

- Serial
  - begin – inițializare port serial (viteză de comunicații) – Serial.begin(9600)
  - print, println – transmiterea unei valori pe port – Serial.println("Arduino")
  - read – recepția unei valori pe port – int a = Serial.read()
- Acces porturi digitale I/O
  - pinMode – setare stare pin digital – pinmode(12,OUTPUT)
  - digitalWrite – scriere pin de ieșire – digitalWrite(12,HIGH)
  - digitalRead – citire pin de intrare – int val = digitalRead(13);

UTILIZAREA LIBRĂRIILOR PENTRU ACCESAREA CIRCUITELOR PERIFERICE

- Acces porturi analogice
  - analogWrite – scriere port "analogic" – de fapt este o comandă PWM! – analogWrite(3,100)
    - Pinul poate fi: 3,5,6,9,10 și 11 !
    - Valoarea scrisă este un număr cuprins între 0 (low mereu) și 255 (high mereu)
    - La alte plăci arduino, de ex. Arduino DUO, există 2 porturi (DAC0 și DAC1) ce permit conversia analogică reală.

    https://www.arduino.cc/reference/en/language/functions/analog-io/analogwrite/

  - analogRead – citire semnal analogic – int val = analogRead(3)
    - Pinii care pot fi intrări analogice sunt: 0,1,2,3,4,5
    - Valoarea citită este între 0 și 1023

    https://www.arduino.cc/reference/en/language/functions/analog-io/analogread/

PROGRAMUL 1

PROGRAM CARE REALIZEAZĂ AFIȘAREA UNUI MESAJ PE CONSOLA ARDUINO – COMUNICAȚIE USB

```
//Programul 1



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

 Serial.println(“**********************************”);
}




void loop() {
  Serial.println("Bine ati venit la cursul 4 de Arduino!!!");
  delay(5000);
}
```

PROGRAMUL 2

PROGRAMUL REALIZEAZA APRINDEREA UNUI LED CONECTAT LA PINUL DIGITAL 13 AL PLACUTEI ARDUINO. LED-UL SE APRINDE ATUNCI CAND PINUL 13 DIGITAL ESTE PUS IN HIGH (DIN PROGRAMUL SCRIS PE ARDUINO) SE STINGE ATUNCI CAND PINUL 13 ESTE PUS IN LOW.

```
//Programul 2

#define led 13

void setup() {
  pinMode(led, OUTPUT);
}
void loop() {
  //Aprind ledul pentru o secunda
  digitalWrite(led, HIGH);
  delay(1000);

  //Sting ledul pentru o secunda
  digitalWrite(led, LOW);
  delay(1000);
}
```

TEMĂ DE CLASĂ

- Aprindeți (ON) și stingeți un LED verde de două ori pe secundă folosind o placă Arduino Uno.
- Afișati textul „Bine ati venit!” pe consola literă cu literă – frecvența de afișare a unei litere este de 0.5 Hz.

VARIABILE. TIPURI DE DATE. CONSTANTE

- **<tip variabilă> <nume variabilă>**
- Declarații variabile:
  - la începutul programului în afara oricărei funcții – variabile globale
  - în program (în funcții) – variabile locale
- Tip variabile:
  - char, byte, int, unsigned int, long, unsigned long, float, double
  - int a; char c; double pret; unsigned int pozitieMotor;
- **#define <nume constanta simbol> <valoare>**
  - **#define MAXIM_PASI 100**

PROGRAMUL 3

PROGRAMUL AFISEAZA IN CONSOLA APLICATIEI ARDUINO IDE, VALOAREA UNUI POTENTIOMETRU CONECTAT LA PINUL ANALOGIC A0 AL PLACUTEI ARDUINO.

```
//Programul 3
int in0=A0;
void setup() {
Serial.begin(9600);
pinMode(in0,INPUT);
}

void loop() {
  int v0=analogRead(in0);
  Serial.print("Valoare potentiometru : ");
  Serial.print(v0);
  Serial.println(" ");
  delay(500);//delay loop
}
```

PROGRAMUL 4

PROGRAMUL REALIZEAZA APRINDEREA SECVENTIALA A UNUI LED RGB

```
//Programul 4

#define r 9
#define g 10
#define b 11

void setup() {
  pinMode(r,OUTPUT);
  pinMode(g,OUTPUT);
  pinMode(b,OUTPUT);
}
void loop() {
 digitalWrite(r,HIGH);
 delay(500);
 digitalWrite(r,LOW);
 digitalWrite(g,HIGH);
 delay(500);
 digitalWrite(g,LOW);
 digitalWrite(b,HIGH);
 delay(500);
 digitalWrite(b,LOW);
}
```

PROGRAMUL 5

PROGRAMUL REALIZEAZA APRINDEREA LED-ULUI RGB INTR-O CULOARE OARECARE, CREATA CU AJUTORUL PINILOR DIGITAL AI ARDUINO, IN CONFIGURATIA PWM.

```
//Programul 5


#define r 9
#define g 10
#define b 11


void setup() {
  pinMode(r,OUTPUT);
  pinMode(g,OUTPUT);
  pinMode(b,OUTPUT);


}
void loop() {
 analogWrite(r,125);
 analogWrite(g,50);
 analogWrite(b,200);
}
```

TEMĂ DE CLASĂ

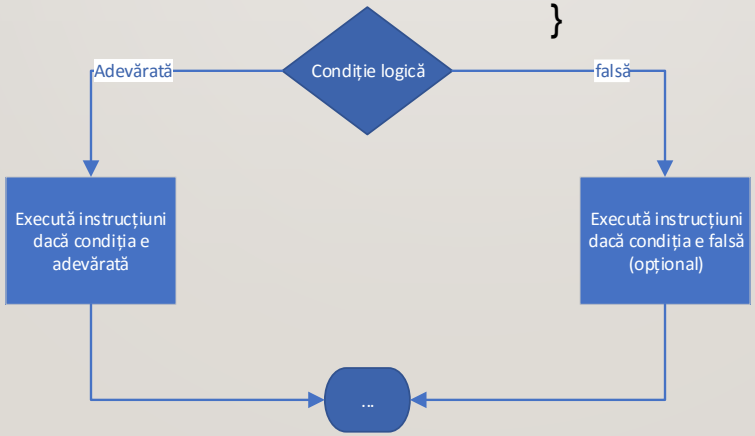
- Puteți face LED-ul multicolor (RGB) să lumineze galben?
- La programul 3 (cu potențiometrul) sunt afișate valori numerice cuprinse între 0 și 1024 care corespund unei tensiuni cuprinse între 0 și 5V. Puteți afișa tensiunea reală pentru orice poziție a potențiometrului?
- Dacă tot ați afișat tensiunea, având un curent constant de 5 mA în potențiometru puteți spune care este rezistența electrică pentru orice poziție (dacă da afișați-o în consolă cu unitatea ei de măsură)

INSTRUCȚIUNI

- Condiționare (if, switch)

if(<condiție logică>) {
<Instrucțiuni dacă condiția e adevărată>
}
[else{
<Instrucțiuni dacă condiția e falsă>
}]

switch(<variabilă testată>)
{
case valoare1 : { <execută instrucțiuni dacă variabila testată este egală cu valoare1 >} break;
……
[default: {<execută instrucțiuni dacă nu e niciuna din valorile testate>} break;]
}



PROGRAMUL 6

PROGRAMUL REALIZEAZA APRINDEREA UNU LED, CONECTAT LA PLACUTA ARDUINO, ATUNCI CAND VALOAREA TENSIUNII DE PE UN POTENTIOMETRU CONECTAT LA PINUL ANALOG A0 AL PLACUTEI, ESTE MAI MARE DECAT UN PRAG PRESETAT.

```
//Programul 6

#define PRAG 500

int in0=A0;

int led=13;

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

pinMode(in0,INPUT);
pinMode(led,OUTPUT);
}

void loop() {
 int v0=analogRead(in0);
 Serial.print("Valoare tensiune potentiometru : ");
 Serial.print(v0);
 Serial.println(" ");
 if(v0>=PRAG)
  digitalWrite(led,HIGH);
 else
  digitalWrite(led,LOW);
 delay(500);//delay loop

}
```

TEMĂ DE CLASĂ

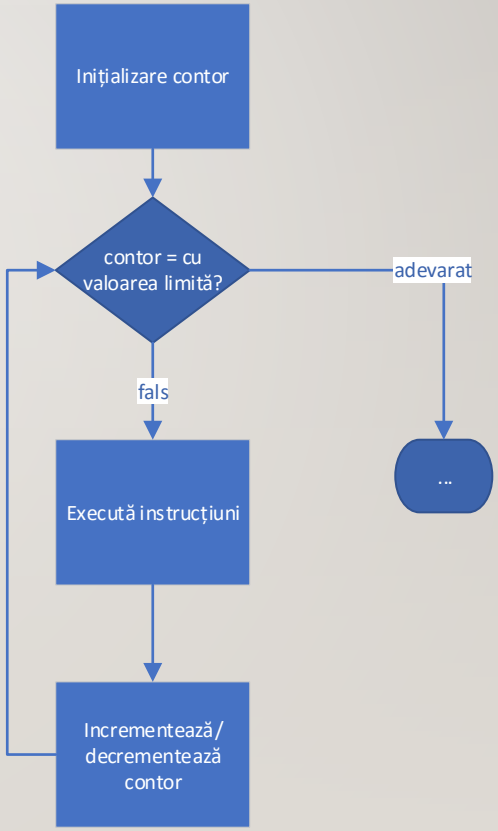
- Montați potențiometrul și LED-ul multicolor. Împărțiți mental potențiometrul în trei părți. Dacă cursorul potențiometrului se află în prima treime aprindeți culoarea roșie, dacă se află în a doua treime aprindeți culoarea verde și dacă se află în ultima treime aprindeți culoarea albastră.

INSTRUCȚIUNI

- Bucle (for, while)

for(<contor>=<valoare inițială>;<comparare contor cu valoarea limită>;incrementare/decrementare contor)
{
<instrucțiuni>
}

while(<condiție logică>)
{
<instrucțiuni>
}



PROGRAMUL 7

GENERAM TREI NUMERE ALEATOARE IN GAMA 0-9, CARE SA FIE DIFERITE INTRE ELE. SA SE AFISEZE NUMERELE

```
//Programul 7

#define GAMA 6

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

}

void loop() {
  int a, b, c;
  a = random(GAMA);
  for (;;) {
    b = random(GAMA);
    if (b != a)break;
  }
  for (;;) {
    c = random(GAMA);
    if (c != a && c != b)break;
  }
  Serial.print("a="); Serial.println(a);
  Serial.print("b="); Serial.println(b);
  Serial.print("c="); Serial.println(c);
  while (1);
}
```

PROGRAMUL 8

GENERAM TREI NUMERE ALEATOARE IN GAMA 0-99. SA SE AFISEZE MAXIMUL DINTRE ACESTEA, IN CONSOLA APLICATIEI ARDUINO IDE

```
//Programul 8

#define GAMA 100

void setup() {

  Serial.begin(9600);

}

void loop() {
 int a=random(GAMA);
 int b=random(GAMA);
 int c=random(GAMA);
int max=a;
if(b>max)max=b;
if(c>max)max=c;
Serial.print("a=");Serial.println(a);
Serial.print("b=");Serial.println(b);
Serial.print("c=");Serial.println(c);
Serial.print("max=");Serial.println(max);
while(1);
}
```

TEMĂ DE CLASĂ

- Faceți un program care să comande LED-ul RGB să treacă prin toate culorile maxim saturate (combinații ale lor) astfel: stins, R, G, RG, B, RB, GB, RGB. Între fiecare lăsați o temporizare de 250 ms.

VECTORI

- **<tip date> <nume vector> <[> <dimensiune 1> <]>**
- **[<[> dimensiune 2 <]> …]**
- byte octetiMesaj[100]; double ponderi[25];
  byte imagine[120][70]
- octetiMesaj[0] = 0x12; byte pixel = imagine[10][10];

PROGRAMUL 9

GENERAM O MATRICE PATRATICA DE DIMENSIUNE N=5, FORMATA DIN NUMERE ALEATOARE IN GAMA 0-99.SA SE AFISEZE MAXIMUL DIN ACEASTA.

```
//Progamul 9
#define GAMA 100
#define N 5
void setup() {
  Serial.begin(9600);
  randomSeed(analogRead(0));
}

void loop() {
  int a[N][N];
  int i, j;
  for (i = 0; i < N; i++)
    for (j = 0; j < N; j++)
      a[i][j] = random(GAMA);
for (i = 0; i < N; i++) {
   for (j = 0; j < N; j++){
     Serial.print(a[i][j]); Serial.print(" ");}
   Serial.println();
 } int max = a[0][0];
 for (i = 0; i < N; i++)
   for (j = 0; j < N; j++)
     if(a[i][j]>max)max=a[i][j];
 Serial.print("max="); Serial.println(max);
   while (1);
}
```

PROGRAMUL 10

GENERAM O MATRICE PATRATICA DE DIMENSIUNE N=20, FORMATA DIN NUMERE ALEATOARE 0 SAU 1. SA SE AFISEZE LINIA CU CELE MAI MULTE VALORI DE 1

```
//Programul 10
#define N 20
void setup() {
  Serial.begin(9600);
  randomSeed(analogRead(0));
}
void loop() {
  int a[N][N];
  int i, j;
  for (i = 0; i < N; i++)
    for (j = 0; j < N; j++)
      a[i][j] = random(2);
  for (i = 0; i < N; i++) {
    for (j = 0; j < N; j++)
Serial.print(a[i][j]);
    Serial.println();
//sa afisam linia care are cele mai multe
//valori de 1
//calculam nr de valori de 1 din prima
//linia:
  int contor=0;
for(i=0;i<N;i++)
    contor=contor+a[0][i];
    //initializam maximul:
    int max=contor;
    int indexLinieMax=0;
    //parcurgem restul linilor:
    for(i=1;i<N;i++){
//calculam cate valori de 1 sunt in linia i:
        contor=0;
for(j=0;j<N;j++)
        contor=contor+a[i][j];
      if(contor>max){
        max=contor;
        indexLinieMax=i;
      }
    }
Serial.print("Nr linie cu cele mai multe
valori de 1 : ");
    Serial.print(indexLinieMax);
    Serial.println();
    for(i=0;i<N;i++)
      Serial.print(a[indexLinieMax][i]);
    Serial.println();
    while (1);
}
```

TEMĂ DE CLASĂ

- Construiti un vector tri-dimensional de dimensiune 100 și populați-l aleatoriu cu numere între 0 și 255. Generați apoi, folosind acest vector, combinații de culori care le trimiteți către LED-ul RGB cu o frecvență de 2Hz.

INSTRUCȚIUNI

- Construire și utilizare funcții

1. Construirea (declararea) unei funcții:
   - <tip returnat> <nume functie> (<lista parametrii>)
   - { <instructiuni> }
2. Utilizarea (apelarea) unei funcții:
   - v = <nume functie> (<parametrii alocati>)
   - Dacă <tip returnat> este void: <nume functie> (<parametrii alocati>)

PROGRAMUL 11

GENERAM TREI NUMERE ALEATOARE IN GAMA 0-99. SA SE CREEZE O FUNCTIE CARE CALCULEAZA MAXIMUL DINTRE ACESTE NUMERE. SA SE AFISEZE MAXIMUL.

```
//Programul 11

#define GAMA 100

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

}

void loop() {
 int a=random(GAMA);
 int b=random(GAMA);
 int c=random(GAMA);
 int rez=maxim(a,b,c);
Serial.print("a=");Serial.println(a);
Serial.print("b=");Serial.println(b);
Serial.print("c=");Serial.println(c);
Serial.print("max=");Serial.println(rez);
while(1);
}
int maxim(int a,int b,int c){

 int max=a;
 if(b>max)max=b;
 if(c>max)max=c;
 return max;

}
```

PROGRAMUL 12

GENERAM UN VECTOR CU N NUMERE ALEATOARE DIFERITE, IN GAMA 0-19.SA CREEZE O FUNCTIE CARE GENEREAZA VECTORUL SI SA SE AFISEZE ACESTA.

```
//Programul 12
#define GAMA 20
#define N 10
void setup() {
  Serial.begin(9600);
}
void loop() {
  int a[N];
  int i;
  generareVector(a,N);

  for(i=0;i<N;i++)
    Serial.println(a[i]);
  while (1);
}
```

```
void generareVector(int a[],int dim){
  a[0]=random(GAMA);
  int i;
  for(i=1;i<dim;i++){
    //generam repetat pe a[i]
    for(;;){
      a[i]=random(GAMA);
      if(estePrezent(a,i,a[i])==false)break;
    }}}
boolean estePrezent(int a[],int nrElemente,int x){
  int i;
  for(i=0;i<nrElemente;i++)
    if(a[i]==x)return true;
  return false;
}
```

PROGRAMUL 13

GENERAM UN VECTOR CARE VA CONTINE PRIMELE N=100 DE NUMERE PRIME. AFISAM APOI ACEST VECTOR.

```
//Programul 13
#define N 100
void setup(){
  Serial.begin(9600);
}
void loop(){
  int a[N];
  int i;
  a[0]=1;
  int nr=2;
  for(i=1;i<N;i++)
    for(;;){
      if(prim(nr)){
        a[i]=nr;
        nr++;
        break;
      }
      else nr++;
    }

  for(i=0;i<N;i++)
    Serial.println(a[i]);
  while (1);
}
boolean prim(int nr){
  int i;
  for(i=2;i<=sqrt(nr);i++)
    if(nr%i==0)return false;
  return true;
}
```

PROGRAMUL 14

VOM SCRIE O FUNCTIE RECURSIVA IN CARE VOM AFISA N=20 STELUTE (CARACTERUL *), PE ORIZONTALA, IN CONSOLA .

```
//Programul 14

#define N 20

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

void loop() {
  afisare(N);
  while (1);
}
void afisare(int nr) {
  if (nr == 1) {
    Serial.print("*");
    return;
  }
  Serial.print("*");
  afisare(nr - 1);
}
```

TEMĂ DE CLASĂ

- Realizați un program care să utilizeze potențiometrul și un LED astfel:
  - La fiecare modificare a valorii potențiometrului să apeleze două funcții care să calculeze tesiunea și rezistența electrică(așa cum am arătat pentru valoarea 1024 tensiunea este de 5V, curentul este constant de 5 mA). Valorile calculate să fie afișate în consolă.
  - Dacă valoarea tensiunii este cuprinsă între 2V – 3V atunci să fie aprins LED-ul.
  - Rata de citire a potențiometrului este de 4 ori pe secundă.
  - Dacă potențiometrul nu este deplasat atunci nu se va afișa nimic în consolă.

INSTRUCȚIUNI

- Structuri de date
  - Reprezintă cele mai simple colecții de date utilizate de obicei pentru a reprezenta variabile cu mai mulți parametrii: de exemplu numere complexe (parte reală și parte imaginară), puncte în plan (coordonata x și y), coordonată geografică (latitudine și longitudine) etc.
  - O tip de variabilă cu mai mulți parametrii se definește ca o structură astfel:
  - **typedef struct { <lista parametrii>} <tip variabila>**

PROGRAMUL 15

FOLOSIND TIPUL DE DATE STRUCTURA, VOM CREA STRUCTURA PUNCT. VOM CALCULA APOI SI AFISA DISTANTA DINTRE DOUA PUNCTE.

```
//Programul 15

typedef  struct {int x;int y;}punct;
void setup() {
  Serial.begin(9600);
}

void loop() {
  punct p;
  p.x = 3;
  p.y = 5;
  double dist = distanta(p);
  Serial.print(dist);
  while(1);
}

double distanta(punct p) {
  return sqrt(p.x * p.x + p.y * p.y);
}
```

TEMĂ DE CLASĂ

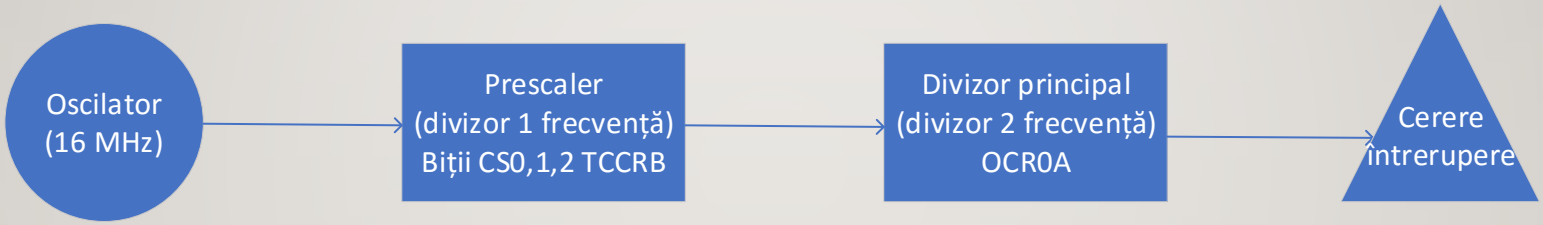
- Definiți pentru LED-ul RGB un tip de dată cu numele „culoare” și cu parametrii R, G, B : intregi. Definiți un vector de 100 de culori, populați-l cu valori aleatoare și apoi afișați-l pe LED la un interval de 1 secundă fiecare culoare.

LUCRU CU ÎNTRERUPERI

- Dispozitivele periferice pot genera cereri de întreruperi;
- La recepția unei astfel de cereri procesorul va suspenda execuția programului și va apela o funcție (subrutină);
- Întreruperi: canale timer (periodice), porturi intrare (evenimente), porturi seriale (recepție date), etc.

LUCRU CU ÎNTRERUPERI

- Schemă bloc canal timer



$$Frecventa\ intrerupere = \frac{frecventa\ oscilator}{divizare\ prescaler\ \times (divizare\ principal + 1)}$$

$$divizare\ principala = \frac{frecventa\ oscilator}{(divizare\ prescaler\ \times frecventa\ intrerupere)} + 1$$

http://www.instructables.com/id/Arduino-Timer-Interrupts/

LUCRU CU ÎNTRERUPERI

- Întreruperi canal timer. Regiștrii de configurare canal timer

TCCR0A – Timer/Counter Control Register A

Bit	7	6	5	4	3	2	1	0	
0x24 (0x44)	COM0A1	COM0A0	COM0B1	COM0B0	–	–	WGM01	WGM00	TCCR0A
Read/Write	R/W	R/W	R/W	R/W	R	R	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Mode	WGM02	WGM01	WGM00	Timer/Counter Mode of Operation	TOP	Update of OCRx at	TOV Flag Set on[1][2]
0	0	0	0	Normal	0xFF	Immediate	MAX
1	0	0	1	PWM, Phase Correct	0xFF	TOP	BOTTOM
2	0	1	0	CTC	OCRA	Immediate	MAX
3	0	1	1	Fast PWM	0xFF	BOTTOM	MAX
4	1	0	0	Reserved	–	–	–
5	1	0	1	PWM, Phase Correct	OCRA	TOP	BOTTOM
6	1	1	0	Reserved	–	–	–
7	1	1	1	Fast PWM	OCRA	BOTTOM	TOP

LUCRU CU ÎNTRERUPERI

- Întreruperi canal timer. Regiștrii de configurare canal timer

TCCR0B – Timer/Counter Control Register B

Bit	7	6	5	4	3	2	1	0	
0x25 (0x45)	FOC0A	FOC0B	–	–	WGM02	CS02	CS01	CS00	TCCR0B
Read/Write	W	W	R	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

CS02	CS01	CS00	Description
0	0	0	No clock source (Timer/Counter stopped)
0	0	1	$clk_{I/O}$/(No prescaling)
0	1	0	$clk_{I/O}$/8 (From prescaler)
0	1	1	$clk_{I/O}$/64 (From prescaler)
1	0	0	$clk_{I/O}$/256 (From prescaler)
1	0	1	$clk_{I/O}$/1024 (From prescaler)
1	1	0	External clock source on T0 pin. Clock on falling edge.
1	1	1	External clock source on T0 pin. Clock on rising edge.

LUCRU CU ÎNTRERUPERI

- Întreruperi canal timer. Regiștrii de configurare canal timer

TIMSK0 – Timer/Counter Interrupt Mask Register

Bit	7	6	5	4	3	2	1	0	
(0x6E)	–	–	–	–	–	OCIE0B	OCIE0A	TOIE0	TIMSK0
Read/Write	R	R	R	R	R	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 2 – OCIE0B: Timer/Counter Output Compare Match B Interrupt Enable**

When the OCIE0B bit is written to one, and the I-bit in the Status Register is set, the Timer/Counter Compare Match B interrupt is enabled. The corresponding interrupt is executed if a Compare Match in Timer/Counter occurs, i.e., when the OCF0B bit is set in the Timer/Counter Interrupt Flag Register – TIFR0.

- **Bit 1 – OCIE0A: Timer/Counter0 Output Compare Match A Interrupt Enable**

When the OCIE0A bit is written to one, and the I-bit in the Status Register is set, the Timer/Counter0 Compare Match A interrupt is enabled. The corresponding interrupt is executed if a Compare Match in Timer/Counter0 occurs, i.e., when the OCF0A bit is set in the Timer/Counter 0 Interrupt Flag Register – TIFR0.

- **Bit 0 – TOIE0: Timer/Counter0 Overflow Interrupt Enable**

When the TOIE0 bit is written to one, and the I-bit in the Status Register is set, the Timer/Counter0 Overflow interrupt is enabled. The corresponding interrupt is executed if an overflow in Timer/Counter0 occurs, i.e., when the TOV0 bit is set in the Timer/Counter 0 Interrupt Flag Register – TIFR0.

PROGRAMUL 16

FOLOSIND ÎNTRERUPEREA DE LA CANALUL TIMER 0 SĂ SE PROGRAMEZE APRINDEREA ȘI STINGEREA UNUI LED LA O SECUNDĂ ÎN TIMP CE ÎN BUCLA PRINCIPALĂ RULEAZĂ O AFIȘARE LA 5 SECUNDE

```
//Programul 16
int contor_ms = 0;
boolean stareLed = false;

#define LED 13

void setup() {

 pinMode(LED, OUTPUT);

 //TCCR0A = 0;
// TCCR0B = 0;
 TCCR0A |= (1 << WGM01); //modul CTC
 TCCR0B |= (1 << CS01) | (1 << CS00);//prescaler 64
 TIMSK0 |= (1 << OCIE0A); //activare intrerupere
 OCR0A = 250;
 //sei();

 Serial.begin(9600);
}

ISR(TIMER0_COMPA_vect){
  contor_ms++;
  if(contor_ms == 1000){
   contor_ms = 0;
   if(stareLed){
     stareLed = false;
     digitalWrite(LED,LOW);
   }
   else{
     stareLed = true;
     digitalWrite(LED,HIGH);
   }
  }
}

void loop() {
    Serial.println("Aici poti afisa si face ce vrei!");
    delay(5000);
}
```

PROGRAMUL 17

REALIZAREA UNUI SISTEM MULTITASKING: TASK-UL 1 ASIGURA COMUTAREA UNUI LED CU O ANUMITĂ FRECVENȚĂ, TASK-UL 2 CITEȘTE POTENȚIOMETRUL ȘI ÎN FUNCȚIE DE VALOAREA LUI SCHIMBĂ FRECVENȚA DE COMUTARE

```
//Programul 17
#define LED 13
int IN0=A0;
int contorMs = 0;
int numarTask = 0;

//task1
int frecventaLED = 1000;
boolean stareLed = false;
int contorMsLED = 0;

//task2
int valoarePotentiometruOld = 0;
int valoarePotentiometruNew = 0;

void setup() {

  pinMode(LED, OUTPUT);
  pinMode(IN0, INPUT);
  TCCR0A |= (1 << WGM01); //modul CTC
  TCCR0B |= (1 << CS01) | (1 << CS00);
  TIMSK0 |= (1 << OCIE0A);
  OCR0A = 250;
  sei();

  Serial.begin(9600);
}

ISR(TIMER0_COMPA_vect){
  if(contorMsLED>0)
    contorMsLED--;

  contorMs++;
  if(contorMs==10){
    contorMs = 0;
    numarTask++;
    if(numarTask == 2)
      numarTask = 0;
  }

}

void task1(){
  if(numarTask==0)
  {
    if(contorMsLED==0){
      contorMsLED = frecventaLED;
      if(stareLed){
        stareLed = false;
        digitalWrite(LED,LOW);
      }
      else{
        stareLed = true;
        digitalWrite(LED,HIGH);
      }
    }
  }
}

void task2(){
  if(numarTask==1){
    valoarePotentiometruNew =
analogRead(IN0);
    if(abs(valoarePotentiometruNew-
valoarePotentiometruOld)>10){
      valoarePotentiometruOld =
valoarePotentiometruNew;
      frecventaLED = valoarePotentiometruNew;
    }
  }
}

void loop() {
 task1();
 task2();

}
```