СОФИЙСКИ УНИВЕРСИТЕТ “СВ. КЛИМЕНТ ОХРИДСКИ”

Факултет по математика и информатика
Специалност: Информатика
Магистърска Програма: Вградени Системи
Асистент: Тома Томов

ПРОЕКТИРАНЕ НА РОБОТИЗИРАНИ СИСТЕМИ

Дисциплина:
Проектиране на роботизирани системи

E-mail:
ttomov75@gmail.com

Тема на упражнението

1. **Предговор**
2. **Променливи**
3. **Деклариране на променливите**
4. **Обхват на променливата**
5. **Масиви**
6. **Аритметика**
7. **Сравнителни оператори**
8. **Логически оператори**
9. **Примери:**

Ардуино

променливи (variables)

Променливата е начин да се именува и да се запази цифрова стойност която да се използва по-късно в програмата. Както предполага името им, променливите могат постоянно да се променят за разлика от константите, чиито стойности никога не се променят. Променливата трябва задължително да се декларира, а задаването на стойността която да се запази е по избор. Кодът долу декларира променлива наречена inputPromenliva и после й задава стойността получена от аналоговия входен пин 2:

```
int inputPromenliva = 0; // deklarira promenliva i

// j zadava stojnost 0

inputPromenliva = analogRead(2); // zadava na promenlivata stojnostta ot

// analogoviya pin 2
```

‘inputPromenliva’ е самата променлива. Първият ред декларира, че тя ще приема стойности от вида integer (int). Вторият ред задава на променливата стойността от аналоговия пин 2. По този начин стойността от пин 2 е достъпна и в други части на кода.

Ардуино

променливи (variables)

След като на променливата е зададена или повторно зададена стойност, може да се провери дали тази стойност отговаря на дадено условие или да се използва направо. Като пример ще покажем три полезни операции с променливи. Следният код проверява дали inputPromenliva е по-малка от 100, и ако условието е вярно задава на променливата стойност 100 и след това задава изчакване (delay) равно на стойността на променливата, която вече е равна поне на 100:

```
if (inputPromenliva < 100) // proveryava dali stojnostta e po-malka ot 100
{
inputPromenliva = 100; // ako gornoto uslovie e vyarno zadava stojnost 100
}
Delay(inputPromenliva); // izpolzva promenlivata za izchakvane (delay)
```

Ардуино

деклариране на променливите (variable declaration)

Всяка променлива трябва да бъде декларирана преди да може да се използва.
Декларирането на променливата означава да се дефинира нейния вид (например integer, long, float, и др.), да й се даде име, и по избор може да и се зададе първоначална стойност. Достатъчно е променливата да се декларира веднъж в програмата, а нейната стойност може да бъде променена по всяко време чрез аритметични функции или други методи на задаване.
Този пример декларира променливата inputPromenliva като integer (int) и й задава първоначална стойност 0.

```
int inputPromenliva = 0;
```

Променливата може да бъде декларирана на различни места в програмата и в зависимост от това къде е дефинирана се определя кои части на програмата могат да я ползват.

Ардуино

Обхват на променливата (variable scope)

Променливата може да бъде декларирана в началото на програмата преди void setup(), локално във функцията и понякога в блок изявления като for циклите. Мястото където е декларирана променливата определя нейния обхват (или възможността на определени части от програмата да я използват).

Глобалната променлива е тази, която може да бъде „видяна” и използвана от всяко изявление или функция на програмата. Такава променлива се дефинира в началото на програмата, преди setup() функцията.

Локалната променлива се дефинира вътре в самата функция или като част от for цикъл. Тя може да се използва само от функцията в която е декларирана. Поради тази причина е възможно да има две или повече променливи с еднакви имена в различни части на програмата и тези променливи да имат различни стойности. За да е по-разбираема програмата и за да се избегне възможността за грешки е добре само една функция да има достъп до стойностите на променливата.

Ардуино

Този пример показва как да се декларират различните видове променливи и техния обхват:

```
int stojnost;          // ’stojnost’ e vidima za
                       // vsyaka funkciya
void setup()
{
//nyama nujda ot setup
}
void loop()
{
for (int i=0; i<20;)   // ’i’ e vidima samo
{                      // za for cikyla
i++;
}
float f;               // ’f’ e vidima samo
}                      // vytre v cikula
```

Ардуино

байт (byte)

Байтът съхранява 8-битова цифрова стойност без десетични запетаи. Използва се за числа от 0 до 255.

```
byte nyakakvaPromenliva = 180;    //deklarira ‘nyakakvaPromenliva’
                                  //kato vid byte
```

цяло число (int)

Целите числа (integer) са основния вид данни за съхранение на числа без десетични запетаи и съхраняват 16-битова стойност за числа от 32,767 до -32,768.

```
int nyakakvaPromenliva = 1500;    // deklarira ‘nyakakvaPromenliva’
                                  // kato vid integer
```

Забележка: Променливите от вид integer ще се ‘превъртят’ ако се преминат минимално или максимално допустимите стойности. Например ако x = 32767 и ако някое изявление добави 1 към x (x = x + 1) тогава x ще се превърти и ще приеме стойност равна на -32,768.

Ардуино

лонг (long)
Това е разширен вид данни за по-дълги integer-и, без десетични запетаи, съхранявани като 32-битови стойности с обхват от 2,147,483,647 до -2,147,483,648.

```
long nyakakvaPromenliva = 90000;      //deklarira ‘nyakakvaPromenliva’
                                      //kato vid long
```

флоут (float)
Вид данни за числа с десетична запетая. Данни от този вид са с по-голяма резолюция от integer, запазват се като 32-битови стойности с обхват от 3.4028235E+38 до -3.4028235E+38.

```
float nyakakvaPromenliva = 3.14;      // deklarira ‘nyakakvaPromenliva’
                                      // kato vid float
```

Забележка: Данните от вид float не са точни и може да дават странни резултати когато бъдат сравнявани. Изчисленията с такива данни протичат много по-бавно отколкото изчисления с числа от вида integer и затова трябва да се избягват винаги когато е възможно.

Ардуино

масиви (arrays)

Масивът е поредица от стойности, достъпни чрез поредните им номера. Всяка стойност в масива може да бъде повикана чрез името на масива и поредния й номер. Масивите са 'zero indexed', което означава, че първата стойност в масива има пореден номер 0. За да може да се използва, масивът трябва да бъде деклариран и по-избор може да му се зададат стойности.

int moyatMasiv [] = {stojnost0, stojnost1, stojnost2…}

Също така е възможно първо да се декларира вида и броя позиции на масива и по-късно да се зададат стойности на съответните позиции:

```
int moyatMasiv [5];       // deklarira masiv ot vid integer s 6 pozicii
moyatMasiv [3] = 10;      // zadava 10 na chetvyrtata stojnost v masiva
```

За да използвате стойност от масива, задайте на някоя променлива името на масива и поредния номер на стойността:

```
x = moyatMasiv [3];       // sega x e raven na 10
```

Ардуино

Масивите често се използват във for цикли, в които „брояч” се използва за да вика
последователно стойностите от масива. Този пример използва масив за да контролира примигването на светодиод. С помощта на for цикъл, броячът започва от 0, задава стойността на позиция 0 в масива ‘primigvane’, в случая 180, на пин 10, изчаква 200ms, и преминава към следващата стойност от масива.

```
int svetodiodPin = 10;                          // svetodiod na pin 10
byte primigvane[] = {180, 30, 255, 200, 10, 90, 150, 60};
                                                // masiv s 8 stojnosti
void setup()
{
pinMode(svetodiodPin, OUTPUT);                  // zadava output pin
}
void loop()
{
for(int i = 0; i < 7; i++)                      // cikylyt syotvetstva na
{                                               // broya pozicii v masiva
analogWrite(svetodiodPin, primigvane[i]);       // zadava nomer na poziciyata
delay(200)                                      // izchakva 200ms
}
}
```

Ардуино

аритметика (arithmetic)
Операторите за аритметика включват събиране, изваждане, умножение и деление. Те връщат сбора, разликата, произведението, или частното на две величини.
y = y + 3;
x = x – 7;
i = j * 6;
r = r / 5;
Операцията се извършва като се използва вида данни на величините които участват в операцията. Така например 9 / 4 ще е равно на 2 вместо на 2.25 тъй като 9 и 4 са от вида integer и не могат да приемат десетични запетаи. Това също значи и че операцията може да „превърти” стойностите ако надвишава възможните стойности за дадения вид данни.
Ако величините са с различен вид данни за операцията се използва по-големия вид данни. Например ако една от величините е от вид float, а другата е от вид integer, за калкулацията ще се използва вида float.

Ардуино

Подбирайте вида данни така че да е достатъчно голям и за най-големите възможни резултати от калкулациите. Помнете при какви стойности става превъртането и какъв ще е резултатът ако числото се превърти. За изчисления които изискват да се смята с дроби използвайте float, но имайте предвид и недостатъците – по-голяма програма и по-бавни калкулации.

Забележка: Използвайте каст оператор (cast operator), примерно (int)moyatFloat за да смените вида данни за величината в движение. Например i = (int)3.6 ще зададе на i стойност равна на 3.

Ардуино

compound assignments

Compound assignments комбинират операторите за аритметика с variable assignment. Те често се срещат във for цикли, както ще покажем по-късно в книжката. Най-често употребяваните compound assignments включват:

```
x++         // syshtoto kato x = x + 1, ili uvelichava x s + 1
x- -        // syshtoto kato x = x - 1, ili namalyava x s - 1
x += y      // syshtoto kato x = x + y, ili uvelichava x s + y
x -= y      // syshtoto kato x = x - y, ili namalyava x s - y
x *= y      // syshtoto kato x = x * y, ili umnojava x s y
x /= y      // syshtoto kato x = x / y, ili deli x s y
```

Забележка: Например x *= 3 ще утрои стойността на x и ще зададе получената стойност на x.

Ардуино

сравнителни оператори (comparison operators)

Сравнението на една променлива или константа с друга често се използва в if заявления за да провери дали дадено условие е истина. В някои от примерите се използва за кое да е, от следните условия:

x = = y // x e raven na y

x ! = y // x ne e raven na y

x < y // x e po-malyk ot y

x > y // x e po-golyam ot y

x <= y // x e po-malyk ili raven na y

x >= y // x e po-golyam ili raven na y

Ардуино

логически оператори (logical operators)

Логическите оператори се използват за да сравнят два израза и като резултат да посочат ‘истина’ или ‘неистина’ (TRUE or FALSE) в зависимост от оператора. Има три логически оператора AND, OR и NOT, които често се използват в if изявления:

Логически AND:

```
if (x > 0 && x < 5)          // ‘istina’ samo ako i dvata
                             // izraza sa verni
```

Логически OR:

```
if (x > 0 | | y > 0)         // ‘istina’ ako koj da e ot
                             // dvata izraza e veren
```

Логически NOT:

```
If (! x > 0)                 // ‘istina’ samo ako
                             // izrazyt ne e veren
```

Ардуино

- Примери:

- регулиране на времето на светене на светодиод с помощта на потенциометър

Ардуино

```
/* Analogov vhod
* ------------
* kontrolira vremeto, za koeto e zapalen svetodiod s pomoshtta na potentsiometur
*/
int potPin = 2;          // vhoden pin za potentsiometura
int ledPin = 13;         // pin za svetodioda
int val = 0;             // promenliva za dannite ot potentsiometura

void setup() {
  pinMode(ledPin, OUTPUT);   // deklarira ledPin kato ishod (OUTPUT)
}

void loop() {
  val = analogRead(potPin);  // otchitane stoynostta ot potentsiometura
  digitalWrite(ledPin, HIGH);  // zapalva svetodioda
  delay(val);                  // spirane na programata za izvestno vreme
  digitalWrite(ledPin, LOW);    // izgasva svetodioda
  delay(val);                  // spirane na programata za izvestno vreme
}
```