

Baza te Informatikes

Leksioni 7

ELISA RECI
Universiteti Luigj Gurakuqi
Fakulteti i Shkencave te Natyres
Departamenti i Matematikes dhe Informatikes
SHKODER

Funksionet ne C++

- Duke përdorur funksionet mund të realizohet strukturimi i programit dhe kështu të shfrytëzohen të gjitha mundësitë për programim të strukturuar që ofron gjuha C++.
- Një funksion është një bllok instruksionesh që ekzekutohet kur thirret nga ndonjë pikë tjetër e programit. Sintaksa e funksionit është:
 - ***tip emer (argument1, argument2, ...) instruksion***
- ku:
 - *tip është tipi i të dhënave që kthehen nga funksioni.*
 - *emer është emri me të cilin do të mund të thirret funksioni.*
 - *argumentX (mund të vendosen sa të duam argumenta).*

Funksionet ne C++

- Çdo argument përbëhet nga një tip të dhënash i ndjekur nga një identifikues, si në deklarin e një variabli (p.sh., int x) dhe ky konsiderohet si një deklarin variabli brenda funksionit, pra variabli i deklaruar si argument është si të gjithë variablat e tjerë brenda funksionit.
- Me anë të argumentave mund ti kalojmë parametra funksionit kur ai thirret.
- Parametrat e ndryshëm ndahen nga njëri-tjetri me presje (,).
- *Instrukcion është trupi i funksionit. Ai mund të jetë një instrukcion i vetëm ose një bllok instruksionesh. Në rastin e fundit duhet të përdoren kllapat {}.*

Funksionet ne C++

```
// shembull funksioni
#include <iostream.h>

int mbledhje (int a, int b)
{
    int r;
    r=a+b;
    return (r);
}

int main ()
{
    int z;
    z = mbledhje (5,3);
    cout << "Rezultati eshte " << z;
    return 0;
}
```

Rezultati eshte 8

Funksionet ne C++

```
// shembull funksioni
#include <iostream.h>

int zbritje (int a, int b)
{
    int r;
    r=a-b;

    return (r);
}

int main ()
{
    int x=5, y=3, z;
    z = zbritje (7,2);
    cout << "Rezultati i pare eshte " << z << '\n';
    cout << "Rezultati i dyte eshte " << zbritje (7,2) << '\n';
    cout << "Rezultati i trete eshte " << zbritje (x,y) << '\n';
    z= 4 + zbritje (x,y);
    cout << "Rezultati i katert eshte " << z << '\n';
    return 0;
}
```

```
Rezultati i pare eshte 5
Rezultati i dyte eshte 5
Rezultati i trete eshte 2
Rezultati i katert eshte 6
```

Funksionet Void

- Thamë se sintaksa e deklarimit të një funksioni është:
- ***tip emer (argument1, argument2, ...) instruksion***
- pra deklarimi i funksionit duhet të fillojë me një ***tip***, që është *tipi i të dhënave që do të kthehen nga funksioni* me anë të instruksionit **return**.
- Si do të veprojmë kur nuk duam të kthejmë asnjë vlerë?
- Le të supozojmë se duam të ndërtojmë një funksion që vetëm paraqet një mesazh në monitor. Nuk duam të kthejmë asnjë vlerë, dhe gjithashtu nuk na duhet asnjë parametër.
- Për këto raste përdoret tipi **void**. **P.sh.:**

Funksionet Void

```
// shembull i nje funksioni void  
#include <iostream.h>  
  
void shkruajmesazh (void)  
{  
    cout << "Mesazh nga funksioni!";  
}  
  
int main ()  
{  
    shkruajmesazh ();  
    return 0;  
}
```

Mesazh nga funksioni!

Funksionet Void

- Megjithëse në C++ nuk është e nevojshme të shkruhet **void**, përdorimi i tij është i përshtatshëm për të treguar se kemi të bëjmë me një funksion pa parametra.
- Është e rëndësishme të theksohet se formati i thirrjes së një funksioni përfshin gjithmonë emrin e tij si dhe argumentat e vendosur ndërmjet kllapave ().
- Edhe kur funksioni nuk ka asnjë argument përdorimi i kllapave është i detyrueshëm. Për këtë arsye thirrja e funksionit **shkruajmesazh** është: **shkruajmesazh ();**
- Në këtë mënyrë sqarohet se kemi të bëjmë me thirrje të një funksioni dhe jo me emrin e një variabli apo diçka tjetër.

Kalimi i parametrave *me vlerë dhe me referencë*

- Deri tani, në të gjithë funksionet e shqyrtuar, kalimi i parametrave tek funksioni është bërë *me vlerë*.
- Kjo do të thotë se kur thërrasim një funksion me parametra, i kalojmë atij vlerat e variablave dhe asnjëherë vetë variablat. P.sh., supozojmë se thërrasim funksionin **mbledhje** duke përdorur kodin e mëposhtëm:
 - `int x=5, y=3, z;`
 - `z = mbledhje (x , y);`
 - Në këtë rast funksioni **mbledhje** thirret duke i kaluar atij vlerat e x dhe y, pra përkatësisht 5 dhe 3,dhe jo vetë variablat.

Kalimi i parametrave *me vlerë dhe me referencë*

- Në këtë mënyrë, kur thirret funksioni mbledhje vlerat e variablave a dhe b bëhen përkatësisht 5 dhe 3, por çdo ndryshim i vlerave të variablit a ose b brenda funksionit mbledhje nuk do të ndikojë në vlerat e variablave x dhe y që janë jashtë tij, pasi funksionit nuk i janë kaluar vetë variablat x dhe y , por vetëm vlerat e tyre.

Kalimi i parametrave *me vlerë dhe me referencë*

- Por mund të ketë raste kur duam të ndryshojmë brenda një funksioni vlerën e një variabli jashtë tij.
- Për këtë qëllim përdoren *parametrat që kalohen me referencë, si në funksionin **dyfishim** të shembullit që vijon:*

Kalimi i parametrave *me vlerë dhe me referencë*

```
// kalimi i parametrave me reference
#include <iostream.h>

void dyfishim (int& a, int& b, int& c)
{
    a*=2;
    b*=2;
    c*=2;
}

int main ()
{
    int x=1, y=3, z=7;
    dyfishim (x, y, z);
    cout << "x=" << x << ", y=" << y << ", z=" << z;
    return 0;
}
```

```
x=2, y=6, z=14
```

Kalimi i parametrave *me vlerë dhe me referencë*

- Së pari vërejmë se në deklarinimin e funksionit dyfishim pas tipit të secilit parametër ka një karakter *ampersand (&)*, që shërben për të treguar se variabli pas tij duhet të kalohet me referencë dhe jo si zakonisht *me vlerë*.
- Kur kalojmë një parametër *me referencë në fakt kalojmë vetë variablin tek funksioni, prandaj çdo* ndryshim që i bëhet parametrin brenda funksionit do të ndikojë në variablin e jashtëm.

Kalimi i parametrave *me vlerë dhe me referencë*

- Në këtë rast parametrat **a, b dhe c** përfaqësojnë brenda funksionit **dyfishim** variablat që janë përdorur në thirrjen e tij (**x, y dhe z**). Prandaj **çdo ndryshim që realizohet tek a brenda funksionit do të paraqitet tek x jashtë tij, e njëjloj çdo ndryshim i b do të ndikojë tek y, dhe i c tek z.**
- Për këtë arsye dalja e programit të dhënë, i cili paraqet në monitor vlerat e variablave **x, y pas z pas** thirrjes së funksionit **dyfishim, tregon vlerat e tre variablave të funksionit main të dyfishuara.**

Kalimi i parametrave *me vlerë dhe me referencë*

- Kalimi i parametrave me referencë është një mënyrë që lejon që një funksion të kthejë më shumë se sa një vlerë. P.sh., më poshtë jepet një funksion i cili kthen numrin paraardhës dhe pasardhës të parametrin të parë që i kalohet.

```
// kthim i me shume se nje vlere
#include <iostream.h>

void parapas (int x, int& para, int& pas)
{
    para = x-1;
    pas = x+1;
}

int main ()
{
    int x=100, y, z;
    parapas (x, y, z);
    cout << "Paraardhesi=" << y << ", Pasardhesi=" << z;
    return 0;
}
```

```
Paraardhesi=99, Pasardhesi=101
```

Vlerat e paracaktuara të parametrave

- Kur deklarohet një funksion është e mundur që të vendoset një vlerë për secilin parametër, që quhet vlerë *e paracaktuar*. *Kjo vlerë do të përdoret në qoftë se parametri përkatës lihet bosh kur thirret funksioni*. Për të realizuar këtë mjafton që ti japim një vlerë parametrut në deklarin e funksionit.
- Në qoftë se vlera për këtë parametër nuk kalohet nga thirrja e funksionit, përdoret vlera e paracaktuar, por në qoftë se nga thirrja kalohet një vlerë atëherë vlera e paracaktuar nuk përdoret.

Vlerat e paracaktuara të parametrave

```
// vlerat e paracaktuara ne nje funksion
#include <iostream.h>

int pjesto (int a, int b=2)
{
    int r;
    r=a/b;
    return (r);
}

int main ()
{
    cout << pjesto (12);
    cout << endl;
    cout << pjesto (20,4);
    return 0;
}
```

```
6
5
```

Funksionet e mbivendosur

- Në C++ është e mundur që dy funksione të kenë të njëjtin emër në qoftë se ato kanë të ndryshme listat e parametrave.
- Kjo do të thotë se dy funksione mund të kenë të njëjtin emër në qoftë se kanë numër të ndryshëm parametrash ose i kanë parametra me tipe të ndryshëm.

Funksionet e mbivendosur

```
// funksionet e mbivendosur
#include <iostream.h>

int pjesto (int a, int b)
{
    return (a/b);
}
```

```
2
2.5
```

```
float pjesto (float a, float b)
{
    return (a/b);
}

int main ()
{
    int x=5,y=2;
    float n=5.0,m=2.0;
    cout << pjesto (x,y);
    cout << "\n";
    cout << pjesto (n,m);
    cout << "\n";
    return 0;
}
```

Ushtrimi 1

- Ndertoni nje program qe nepermjet nje funksioni rendit ne rendin rrites tre nr te dhene.

Ushtrimi 2

- Te ndertohet nje program qe perdore nje funksion per llogaritjen e sip tedrejtekeshit dhe katrorit.
- Te krijohen funksione te mbivendosura
- `llogarit(int a, int b)`
- `llogarit(double a, double b)`

Ushtrimi 3

- Ndertoni nje program ne C++ qe nderthur perdorimin e funksioneve
- Ku njeri funksion afishon mesazhin si me poshte

```
=====
UNIVERSITETI LUIGJ GURAKUQI
Fakulteti i Shkencave te Natyres
Dega Informatik
=====
```

- Si dhe daten dhe oren.
- Ndersa funksioni tjeter realizon llogaritjen e teoremes se pitagores

Ushtrimi 4

- Implementoni ne paskal program qe gjen rrenjet katrore te ekuacionit: $ax^2 + bx + c = 0$

Ushtrimi 5

- Ndertoni nje program qe llogarit peshen ideale te nje personi nepermjet formules $P=(GJ-100 + M/10)*KM$, ku $k_m=0.9$, $k_f=0.8$
- Ku gj-gjinia;
- K_m , k_f - koeficientet per mashkullin dhe femren

