BİL106 Nesne Yönelimli Programlama

Dr. Öğr. Üyesi Yavuz CANBAY

Kahramanmaraş Sütçü İmam Üniversitesi

Bilgisayar Mühendisliği Bölümü

Bölüm 5: Fonksiyonlar

Fonksiyonlar

- Programın kavramsal olarak düzenlenmesine yardımcı olan yapılardır
- Programın boyutunu küçülterek karmaşıklığın önüne geçilmesini sağlar

Basit Fonksiyonlar

```
// table.cpp
// demonstrates simple function
#include <iostream>
using namespace std;

void starline();                        //function declaration
                                        //   (prototype)
int main()
   {
   starline();                          //call to function
   cout << "Data type   Range" << endl;
   starline();                          //call to function
   cout << "char        -128 to 127" << endl
        << "short       -32,768 to 32,767" << endl
        << "int         System dependent" << endl
        << "long        -2,147,483,648 to 2,147,483,647" << endl;
   starline();                          //call to function
   return 0;
   }
//--------------------------------------------------------------
// starline()
// function definition
void starline()                         //function declarator
   {
   for(int j=0; j<45; j++)              //function body
      cout << '*';
   cout << endl;
   }
```

Fonksiyonlara Parametre Göndermek-sabit

```
// tablearg.cpp
// demonstrates function arguments
#include <iostream>
using namespace std;
void repchar(char, int);                //function declaration

int main()
   {
   repchar('-', 43);                    //call to function
   cout << "Data type   Range" << endl;
   repchar('=', 23);                    //call to function
   cout << "char        -128 to 127" << endl
        << "short       -32,768 to 32,767" << endl
        << "int         System dependent" << endl
        << "double      -2,147,483,648 to 2,147,483,647" << endl;
   repchar('-', 43);                    //call to function
   return 0;
   }
//--------------------------------------------------------------
// repchar()
// function definition
void repchar(char ch, int n)            //function declarator
   {
   for(int j=0; j<n; j++)               //function body
      cout << ch;
   cout << endl;
   }
```

Fonksiyonlara Parametre Göndermek-değişken

```
// vararg.cpp
// demonstrates variable arguments
#include <iostream>
using namespace std;
void repchar(char, int);                //function declaration

int main()
   {
   char chin;
   int nin;

   cout << "Enter a character: ";
   cin >> chin;
   cout << "Enter number of times to repeat it: ";
   cin >> nin;
   repchar(chin, nin);
   return 0;
   }
//-------------------------------------------------------------
// repchar()
// function definition
void repchar(char ch, int n)            //function declarator
   {
   for(int j=0; j<n; j++)               //function body
      cout << ch;
   cout << endl;
   }
```

Fonksiyonlara Parametre Göndermek-yapı

```
// engldisp.cpp
// demonstrates passing structure as argument
#include <iostream>
using namespace std;
////////////////////////////////////////////////////////////////
struct Distance              //English distance
   {
   int feet;
   float inches;
   };
////////////////////////////////////////////////////////////////
void engldisp( Distance );     //declaration

int main()
   {
   Distance d1, d2;            //define two lengths

                               //get length d1 from user
   cout << "Enter feet: ";  cin >> d1.feet;
   cout << "Enter inches: ";  cin >> d1.inches;

                               //get length d2 from user
   cout << "\nEnter feet: ";  cin >> d2.feet;
   cout << "Enter inches: ";  cin >> d2.inches;

   cout << "\nd1 = ";
   engldisp(d1);               //display length 1
   cout << "\nd2 = ";
   engldisp(d2);               //display length 2
   cout << endl;
   return 0;
   }
//--------------------------------------------------------------
// engldisp()
// display structure of type Distance in feet and inches
void engldisp( Distance dd )   //parameter dd of type Distance
   {
   cout << dd.feet << "\'-" << dd.inches << "\"";
   }
```

Fonksiyonlarda Değer Döndürmek

```
// convert.cpp
// demonstrates return values, converts pounds to kg
#include <iostream>
using namespace std;
float lbstokg(float);   //declaration

int main()
   {
   float lbs, kgs;

   cout << "\nEnter your weight in pounds: ";
   cin >> lbs;
   kgs = lbstokg(lbs);
   cout << "Your weight in kilograms is " << kgs << endl;
   return 0;
   }
//-------------------------------------------------------------
// lbstokg()
// converts pounds to kilograms
float lbstokg(float pounds)
   {
   float kilograms =  0.453592 * pounds;
   return kilograms;
   }
```

Fonksiyonlarda Yapı Değişkenlerini Döndürmek

```
// retstrc.cpp
// demonstrates returning a structure
#include <iostream>
using namespace std;
////////////////////////////////////////////////////////////////
struct Distance                         //English distance
   {
   int feet;
   float inches;
   };
////////////////////////////////////////////////////////////////
Distance addengl(Distance, Distance);  //declarations
void engldisp(Distance);

int main()
   {
   Distance d1, d2, d3;                 //define three lengths
                                        //get length d1 from user
   cout << "\nEnter feet: ";  cin >> d1.feet;
   cout << "Enter inches: ";  cin >> d1.inches;
                                        //get length d2 from user
   cout << "\nEnter feet: ";  cin >> d2.feet;
   cout << "Enter inches: ";  cin >> d2.inches;

   d3 = addengl(d1, d2);                //d3 is sum of d1 and d2
   cout << endl;
   engldisp(d1); cout << " + ";         //display all lengths
   engldisp(d2); cout << " = ";
   engldisp(d3); cout << endl;
   return 0;
   }
//--------------------------------------------------------------
// addengl()
// adds two structures of type Distance, returns sum
Distance addengl( Distance dd1, Distance dd2 )
   {
   Distance dd3;                //define a new structure for sum

   dd3.inches = dd1.inches + dd2.inches;  //add the inches
   dd3.feet = 0;                          //(for possible carry)
   if(dd3.inches >= 12.0)                 //if inches >= 12.0,
      {                                   //then decrease inches
      dd3.inches -= 12.0;                 //by 12.0 and
      dd3.feet++;                         //increase feet
      }                                   //by 1
   dd3.feet += dd1.feet + dd2.feet;       //add the feet
   return dd3;                            //return structure
   }
//--------------------------------------------------------------
// engldisp()
// display structure of type Distance in feet and inches
void engldisp( Distance dd )
   {
   cout << dd.feet << "\'-" << dd.inches << "\"";
   }
```