Dynamic Initialization of Objects Using Constructors | C++ Tutorials for Beginners #33

Dynamic Initialization of Objects Using Constructors | C++ Tutorials for Beginners #33

In this tutorial, we will discuss the dynamic initialization of objects using constructors in C++

Dynamic Initialization of Objects Using Constructors

The dynamic initialization of the object means that the object is initialized at the runtime. Dynamic initialization of the object using a constructor is beneficial when the data is of different formats. An example program is shown below to demonstrate the concept of dynamic initialization of objects using constructors.

```
#include<iostream>
using namespace std;
```

```
class BankDeposit{
    int principal;
    int years;
    float interestRate;
    float returnValue;

    public:
        BankDeposit(){}
        BankDeposit(int p, int y, float r); // r can be a value like 0.04
        BankDeposit(int p, int y, int r); // r can be a value like 14
        void show();
};
```

Code Snippet 1: Dynamic Initialization of Objects using Constructor Example

As shown in Code Snippet 1,

- 1st we created a “BankDeposit” class which consists of private data members “principal”, “years”, “interestRate”, and “returnValue”.
- 2nd default constructor of the “BankDeposit” class is declared.
- 3rd parameterized constructor of the “BankDeposit” class is declared which takes three parameters “p”, “y”, and “r”. The main thing to note here is that the parameter “r” is of a float data type.
- 4th parameterized constructor of the “BankDeposit” class is declared which takes three parameters “p”, “y”, and “r”. The main thing to note here is that the parameter “r” is of an integer data type.
- 5th function “show” is declared.

The definition of constructors and function is shown below.

```
BankDeposit :: BankDeposit(int p, int y, float r)
{
    principal = p;
    years = y;
    interestRate = r;
    returnValue = principal;
    for (int i = 0; i < y; i++)
    {
        returnValue = returnValue * (1+interestRate);
    }
}

BankDeposit :: BankDeposit(int p, int y, int r)
{
    principal = p;
    years = y;
    interestRate = float(r)/100;
    returnValue = principal;
    for (int i = 0; i < y; i++)
    {
        returnValue = returnValue * (1+interestRate);
    }
}
```

```
void BankDeposit :: show(){
    cout<<endl<<"Principal amount was "<<principal
        << ". Return value after "<<years
        << " years is "<<returnValue<<endl;
}
```

Code Snippet 2: Definition of Constructors and Function

As shown in Code snippet 2,

- 1st the constructor “BankDeposit” is defined in which the value of the parameter “p” is assigned to the data member “principal”; the value of the parameter “y” is assigned to the data member “year”; the value of the parameter “r” is assigned to the data member “interestRate”. At the end “for” loop is defined which will run till the length of the variable “y” and add “1” in the “interestRate”; then multiply the value with the “returnValue”. The main thing to note here is that in this constructor the data type of the parameter “r” is float.
- 2nd another constructor “BankDeposit” is defined in which the value of the parameter “p” is assigned to the data member “principal”; the value of the parameter “y” is assigned to the data member “year”; the value of the parameter “r” is converted to “float” and divided by “100” then assigned to the data member “interestRate”. At the end “for” loop is defined which will run till the length of the variable “y” and add “1” in the “interestRate”; then multiply the value with the “returnValue”. The main thing to note here is that in this constructor the data type of the parameter “r” is float.
- 3rd the function “show” is defined which will print the values of the data members “principal”, “year”, and “returnValue”.

The main program is shown in code snippet 3.

```
int main(){
    BankDeposit bd1, bd2, bd3;
```

```
    int p, y;
    float r;
    int R;


    cout<<"Enter the value of p y and r"<<endl;
    cin>>p>>y>>r;
    bd1 = BankDeposit(p, y, r);
    bd1.show();

    cout<<"Enter the value of p y and R"<<endl;
    cin>>p>>y>>R;
    bd2 = BankDeposit(p, y, R);
    bd2.show();
    return 0;
}
```

Code Snippet 3: Main Program

As shown in a code snippet 3,

- 1st the object “bd1”, “bd2”, and “bd3” of the data type “BankDeposit” are created.
- 2nd the integer variables “p” and “y” are declared; the float variable “r” is declared, and the integer variable “R” is declared.
- 3rd the values for the variables “p”, “y”, and”r” are taken from the user on the runtime.
- 4th parameterized constructor “BankDeposit” is called with the object “bd1” and the variables “p”, “y”, and “r” are passed. The main thing to note here is that this will run the constructor with float parameters “r”.

- 5th function “show” is called which will print the values of data members
- 6th the values for the variables “p”, “y”, and ”R” are taken from the user on the runtime.
- 7th parameterized constructor “BankDeposit” is called with the object “bd2” and the variables “p”, “y”, and “R” are passed. The main thing to note here is that this will run the constructor with integer parameters “R”.
- 8th function “show” is called which will print the values of data members.

The output for the following program is shown in figure 1.

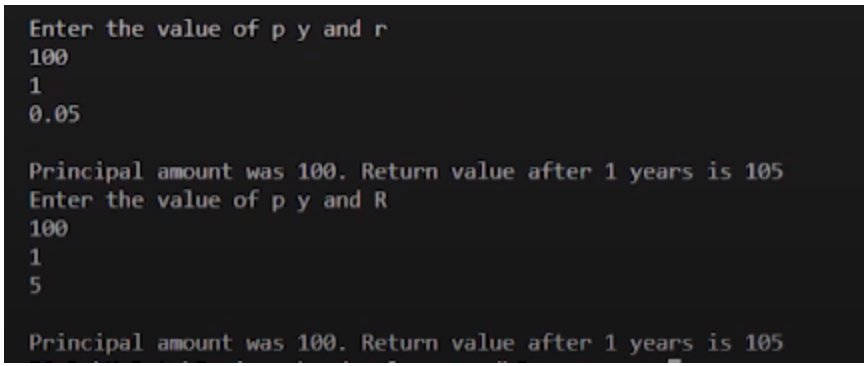

Figure 1: Program Output

As shown in figure 1, the first time the values “100”, “1”, and “0.05” are entered and it gives us the return value of “105”. The second time the values “100”, “1”, and “5” are entered and it gives us the return value of “105”. So the main thing to note here is that the compiler figures out the run time by seeing the data type and runs the relevant constructor.

Previous Next