FAST NATIONAL UNIVERSTIY OF COMPUTER AND EMERGING SCIENCES, PESHAWAR

DEPARTMENT OF COMPUTER SCIENCE

CL217 – OBJECT ORIENTED PROGRAMMING LAB

LAB MANUAL # 09

INSTRUCTOR: MUHAMMAD ABDULLAH

SEMESTER SPRING 2021

OBJECT ORIENTED PROGRAMMING LANGUAGE

Table of Contents

Classes Constructor

A **class constructor** is a special member function of a **class** that is executed whenever we create new objects of that **class**. A **constructor** will have exact same name as the **class** and it does not have any return type at all, not even void.

Default Constructor

Given example demonstrates the concept of default constructor.

```
#include <string>
#include <iostream>
using namespace std;
class ClassRoom {
private:
int roomID;
int numberOfChairs;
char boardType; // C for chalk and M for marker
string multimedia;
string remarks;
public:
	ClassRoom();
	void setroomID(int);
	void setnumberOfChairs(int);
	void setboardType(char);
	void setmultimedia(string);
	void setremarks(string);
	int getroomID();
	int getnumberOfChairs();
	char getboardType();
	string getmultimedia();
	string getremarks();
	void display();
};
void main ()
{
	ClassRoom CR0;
	CR0.display();
	system("PAUSE");
}
//--------------------------------------Constructors
ClassRoom::ClassRoom()
{
	this->roomID=0;
	this->numberOfChairs=0;
```

```
      this->boardType='M';
      this->multimedia="New";
      this->remarks="Default Constructor";
}
//-------------------------------------Getter Functions
int ClassRoom::getroomID()
{
      return this->roomID;
}
int ClassRoom::getnumberOfChairs()
{
      return this->numberOfChairs;
}
char ClassRoom::getboardType()
{
      return (this->boardType);
}
string ClassRoom::getmultimedia()
{
      return this->multimedia;
}

string ClassRoom::getremarks()
{
      return this->remarks;
}
//-----------------------------------Printing Functions
void ClassRoom::display()
{
      cout<<"Room ID \t: "<<this->getroomID()<<endl;
      cout<<"N.O. Chairs \t: "<<this->getnumberOfChairs()<<endl;
      cout<<"Board Type \t: "<<this->getboardType()<<endl;
      cout<<"Multimedia \t: "<<this->getmultimedia()<<endl;
      cout<<"Remarks \t: \n"<<this->getremarks()<<endl;
      cout<<"----------------------------------"<<endl;
}

/*
Room ID         : 0
N.O. Chairs     : 0
Board Type      : M
Multimedia      : New
Remarks         :
Default Constructor
----------------------------------
Press any key to continue . . .
*/
```

Parameterized Constructor

Given example demonstrates the concept of parameterized constructor.

```
#include <string>
#include <iostream>
using namespace std;
class ClassRoom {
private:
int roomID;
int numberOfChairs;
char boardType; // C for chalk and M for marker
string multimedia;
string remarks;
public:
	ClassRoom(int id,int NOC,char,string mul, string rmks);
	void setroomID(int);
	void setnumberOfChairs(int);
	void setboardType(char);
	void setmultimedia(string);
	void setremarks(string);
	int getroomID();
	int getnumberOfChairs();
	char getboardType();
	string getmultimedia();
	string getremarks();
	void display();
};
void main ()
{
	ClassRoom CR(11,25,'M',"Repairing..","Remarks added from Main");
	CR.display();
	system("PAUSE");
}
//--------------------------------------Constructors
ClassRoom::ClassRoom(int id,int NOC,char c,string mul, string remks)
{
	this->roomID=id;
	this->numberOfChairs=NOC;
	this->boardType=c;
	this->setmultimedia(mul);
	this->setremarks(remks+"\nConstructor with All(5)Parameters
(ID,NOC,BoardType,Multimedia,Remarks)");
}
//---------------------------------------Setter Functions
void ClassRoom::setroomID(int id)
```

```
{
	this->roomID=id;
}
void ClassRoom::setnumberOfChairs(int NOC)
{
	this->numberOfChairs=NOC;
}
void ClassRoom::setboardType(char c)
{
	this->boardType=c;
}
void ClassRoom::setmultimedia(string str)
{
	this->multimedia=str;
}
void ClassRoom::setremarks(string str)
{
	this->remarks=str;
}
//--------------------------------------Getter Functions
int ClassRoom::getroomID()
{
	return this->roomID;
}
int ClassRoom::getnumberOfChairs()
{
	return this->numberOfChairs;
}
char ClassRoom::getboardType()
{
	return (this->boardType);
}
string ClassRoom::getmultimedia()
{
	return this->multimedia;
}

string ClassRoom::getremarks()
{
	return this->remarks;
}
//------------------------------------Printing Functions
void ClassRoom::display()
{
	cout<<"Room ID \t: "<<this->getroomID()<<endl;
	cout<<"N.O. Chairs \t: "<<this->getnumberOfChairs()<<endl;
	cout<<"Board Type \t: "<<this->getboardType()<<endl;
	cout<<"Multimedia \t: "<<this->getmultimedia()<<endl;
	cout<<"Remarks \t: \n"<<this->getremarks()<<endl;
	cout<<"----------------------------------"<<endl;
}
/*
Room ID         : 11
N.O. Chairs     : 25
Board Type      : M
Multimedia      : Repairing..
Remarks         :
Remarks added from Main
Constructor with All(5)Parameters
(ID,NOC,BoardType,Multimedia,Remarks)
----------------------------------
Press any key to continue . . .


*/
```

Overloaded Constructor

Given example demonstrates the concept of overloaded constructor

```
#include <string>
#include <iostream>
using namespace std;
class ClassRoom {
private:
int roomID;
int numberOfChairs;
char boardType; // C for chalk and M for marker
string multimedia;
string remarks;
public:
ClassRoom();
ClassRoom(int id);
ClassRoom(int id,int NOC);
ClassRoom(int id,int NOC,char c);
ClassRoom(int id,int NOC,char,string mul);
ClassRoom(int id,int NOC,char,string mul, string rmks);
void ClassRoom::display()
{
cout<<"Room ID \t: "<<this->getroomID()<<endl;
cout<<"N.O. Chairs \t: "<<this->getnumberOfChairs()<<endl;
cout<<"Board Type \t: "<<this->getboardType()<<endl;
cout<<"Multimedia \t: "<<this->getmultimedia()<<endl;
cout<<"Remarks \t: \n"<<this->getremarks()<<endl;
cout<<"----------------------------------"<<endl;
}
};
void main ()
```

```
{
ClassRoom CR0,CR1(1),CR2(2,30),CR3(3,35,'M'),
CR4(4,40,'M',"Repairing.."),CR5(5,40,'M',"Repairing..","Remarks added from Main");
CR0.display();
CR1.display();
CR2.display();
CR3.display();
CR4.display();
CR5.display();
system("PAUSE");
}
//--------------------------------------Constructors
ClassRoom::ClassRoom()
{
       this->setmultimedia("New");
this->remarks="Constructor with 1-Parameter (ID) ";
}
ClassRoom::ClassRoom(int id,int NOC)
{
this->roomID=id;
this->numberOfChairs=NOC;
this->setboardType('M');
this->setmultimedia("New");
this->remarks="Constructor with 2-Parameters (ID,NOC) ";
}
ClassRoom::ClassRoom(int id,int NOC,char c)
{
this->roomID=id;
this->numberOfChairs=NOC;
this->boardType=c;
this->setmultimedia("New");
this->remarks="Constructor with 3-Parameters (ID,NOC,BoardType) ";
}
ClassRoom::ClassRoom(int id,int NOC,char c,string mul)
{
this->setroomID(id); //this->roomID=id;
this->setnumberOfChairs(NOC); //this->numberOfChairs=NOC;
this->setboardType(c); //this->boardType=c;
this->setmultimedia(mul); //this->multimedia=str;
this->setremarks("Constructor with 4-Parameters (ID,NOC,BoardType,Multimedia)");
}
ClassRoom::ClassRoom(int id,int NOC,char c,string mul, string remks)
{
this->setroomID(id); //this->roomID=id;
this->setnumberOfChairs(NOC); //this->numberOfChairs=NOC;
this->setboardType(c); //this->boardType=c;
this->setmultimedia(mul); //this->multimedia=str;
this->setremarks(remks+"\nConstructor with All(5)Parameters
(ID,NOC,BoardType,Multimedia,Remarks)");
}
}
this->roomID=0;
this->numberOfChairs=0;
this->boardType='M';
this->multimedia="New";
this->remarks="Default Constructor";
}
ClassRoom::ClassRoom(int id)
```

```
{
this->roomID=id;
this->setnumberOfChairs(0);
this->setbo          rdType('M');
this->setmultimedia("New");
this->remarks="Constructor with 1-Parameter (ID) ";
}
ClassRoom::ClassRoom(int id,int NOC)
{
this->roomID=id;
this->numberOfChairs=NOC;
this->setboardType('M');
this->setmultimedia("New");
this->remarks="Constructor with 2-Parameters (ID,NOC) ";
}
ClassRoom::ClassRoom(int id,int NOC,char c)
{
this->roomID=id;
this->numberOfChairs=NOC;
this->boardType=c;
this->setmultimedia("New");
this->remarks="Constructor with 3-Parameters (ID,NOC,BoardType) ";
}
ClassRoom::ClassRoom(int id,int NOC,char c,string mul)
{
this->setroomID(id); //this->roomID=id;
this->setnumberOfChairs(NOC); //this->numberOfChairs=NOC;
this->setboardType(c); //this->boardType=c;
this->setmultimedia(mul); //this->multimedia=str;
this->setremarks("Constructor with 4-Parameters (ID,NOC,BoardType,Multimedia)");
}
ClassRoom::ClassRoom(int id,int NOC,char c,string mul, string remks)
{
this->setroomID(id); //this->roomID=id;
this->setnumberOfChairs(NOC); //this->numberOfChairs=NOC;
this->setboardType(c); //this->boardType=c;
this->setmultimedia(mul); //this->multimedia=str;
this->setremarks(remks+"\nConstructor with All(5)Parameters
(ID,NOC,BoardType,Multimedia,Remarks)");
}
/*
Room ID : 0
N.O. Chairs : 0
Board Type : M
Multimedia : New
Remarks :
Default Constructor
----------------------------------
Room ID : 1
N.O. Chairs : 0
Board Type : M
Multimedia : New
Remarks :
Constructor with 1-Parameter (ID)
----------------------------------
Room ID : 2
N.O. Chairs : 30
Board Type : M
```

```
Multimedia : New
Remarks :
Constructor with 2-Parameters (ID,NOC)
----------------------------------
Room ID : 3
N.O. Chairs : 35
Board Type : M
Multimedia : New
Remarks :
Constructor with 3-Parameters (ID,NOC,BoardType)
----------------------------------
Room ID : 4
N.O. Chairs : 40
Board Type : M
Multimedia : Repairing..
Remarks :
Constructor with 4-Parameters (ID,NOC,BoardType,Multimedia)
----------------------------------
Room ID : 5
N.O. Chairs : 40
Board Type : M
Multimedia : Repairing..
Remarks :
Remarks added from Main
Constructor with All(5)Parameters (ID,NOC,BoardType,Multimedia,Remarks)
----------------------------------
Press any key to continue . . .
*/
```

Constructor with Default Parameters

Given example demonstrates the concept of constructor with default values

```
#include <string>
#include <iostream>
using namespace std;
class ClassRoom {
private:
	int roomID;
	int numberOfChairs;
	char boardType; // C for chalk and M for marker
	string multimedia;
	string remarks;
public:
	ClassRoom(int id=0,int NOC=25,char c='C',string mul="New",string
rmks="Defaut Value");
	void setroomID(int);
	void setnumberOfChairs(int);
	void setboardType(char);
	void setmultimedia(string);
	void setremarks(string);
	int getroomID();
```

```
      int getnumberOfChairs();
      char getboardType();
      string getmultimedia();
      string getremarks();
      void display();
};
void main ()
{
      ClassRoom CR0,CR1(1),CR2(2,30),CR3(3,35,'M'),
      CR4(4,40,'M',"Repairing.."),CR5(5,40,'M',"Repairing..","Remarks
added from Main");
      CR0.display();
      CR1.display();
      CR2.display();
      CR3.display();
      CR4.display();
      CR5.display();
      system("PAUSE");
}

//--------------------------------------Constructor
ClassRoom::ClassRoom(int id,int NOC,char c,string mul, string remks)
{
this->setroomID(id); //this->roomID=id;
this->setnumberOfChairs(NOC); //this->numberOfChairs=NOC;
this->setboardType(c); //this->boardType=c;
this->setmultimedia(mul); //this->multimedia=str;
this->setremarks(remks);
}
// All Setter & Getter Functions Deffinitions <Code here all the
setter and getter functions>
//Printing Functions <code here all printing functions>
/*
Room ID : 0
N.O. Chairs : 0
Board Type : M
Multimedia : New
Remarks :
Default Constructor
----------------------------------
Room ID : 1
N.O. Chairs : 0
Board Type : M
Multimedia : New
Remarks :
Constructor with 1-Parameter (ID)
```

```
----------------------------------
Room ID : 2
N.O. Chairs : 30
Board Type : M
Multimedia : New
Remarks :
Constructor with 2-Parameters (ID,NOC)
----------------------------------
Room ID : 3
N.O. Chairs : 35
Board Type : M
Multimedia : New
Remarks :
Constructor with 3-Parameters (ID,NOC,BoardType)
----------------------------------
Room ID : 4
N.O. Chairs : 40
Board Type : M
Multimedia : Repairing..
Remarks :
Constructor with 4-Parameters (ID,NOC,BoardType,Multimedia)
----------------------------------
Room ID : 5
N.O. Chairs : 40
Board Type : M
Multimedia : Repairing..
Remarks :
Remarks added from Main
Constructor with All(5)Parameters
(ID,NOC,BoardType,Multimedia,Remarks)
----------------------------------
Press any key to continue . . .
*/
```