Object Oriented Programming

Classes

Mr. Usman Wajid

usman.wajid@nu.edu.pk

National University
of Computer & Emerging Sciences

Procedural Vs Object Oriented programming

Procedural programming

This approach is populated with variable of different types (data) and the grouped pieces of code (functions)

- It works well for simple and small project

Object Oriented Programming

The data and functions are enclosed together in the same environment

- It is suitable for large and complex projects
- C++ was created as a universal tool for object orient programming

Syntax of a Class

- The members of a class are private (access specifier) by default

```
class <identifier>{
private:
        <type> <identifier_list>;
        <type> <identifier_list>;
        ...
public:
        <type> <function 1>{
        }
        <type> <funtion 2>{
        }
        ...
};
```

An example

```
class Student {

        private:
                int rollNo=0;
                string name="N/A";
                char section='-';
        public:
                void display();
 };
```

- here the *private* and *public* keywords are called access specifiers
- the variable are called data members
- the functions are called functions or operations of the class

An example continued ...

```
void Student::display(){
        cout<<"roll no: "<<rollNo<<endl;
        cout<<"name: "<<name<<endl;
        cout<<"section: "<<section<<endl;
}
```

- a function declared inside the class body can be defined outside the class body using the :: (*scope resolution operator*)

An example continued ...

```
int main() {

        Student ali:
        ali.display();

        return 0;

}
```

An example continued ...

```
class Student {

        private:
                int rollNo=0;
                string name="N/A";
                char section='-';
        public:
                void display();
};

void Student::display(){
        cout<<"roll no: "<<rollNo<<endl;
        cout<<"name: "<<name<<endl;
        cout<<"section: "<<section<<endl;
}

int main() {

        Student ali;
        ali.display();

        return 0;

}
```

Constructors in Class

Constructor

A constructor is a function with a same name as the class but with no return type. It is automatically called when a object of a class is created. It has to be a public access specifier

```
class Student {
        private:
                int rollNo=0;
                string name="N/A";
                char section='-';
        public:
                Student(int x, string y, char b){
                        rollNo = x;
                        name = y;
                        section = b;
                }
                void display();
};
```

Constructors in Class

```
int main() {

        Student ali(1,"ali imran", 'A');
        ali.display();

        return 0;

}
```

Constructors in Class

```
class Student {

        private:
                int rollNo=0;
                string name="N/A";
                char section='-';
        public:
                Student(int x, string y, char b){
                        rollNo = x;
                        name = y;
                        section = b;
                }
                void display();
};

void Student::display(){
        cout<<"roll no: "<<rollNo<<endl;
        cout<<"name: "<<name<<endl;
        cout<<"section: "<<section<<endl;
}

int main() {

        Student ali(1,"ali imran", 'A');
        ali.display();

        return 0;

}
```