Digital Assignment-1

Object Oriented Programming (Theory)

Submitted by: Hari Krishna Shah

VIT ID: 21BCS0167

Google Drive Link is attached below:

https://drive.google.com/drive/folders/1ASv5Z77G6VQfwjvY5fJButeN4qeh3VYv?usp=sharing

1. Construct an OOP to find the distance travelled by a Vehicle in "t" seconds is given by

Distance = u x t = ((a x t2) / 2).

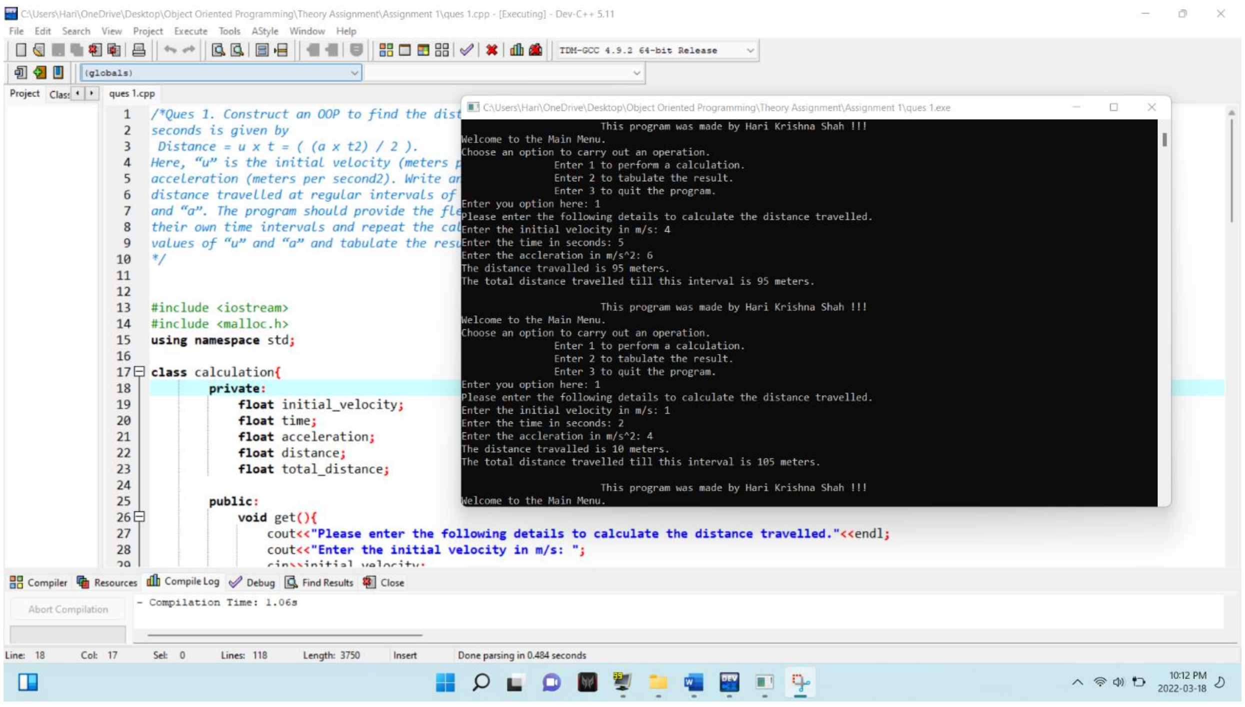
Here, "u" is the initial velocity (meters per second) and "a" is the acceleration (meters per second2). Write an OOP to evaluate the distance travelled at regular intervals of time, given the values of "u" and "a". The program should provide the flexibility to the user to select their own time intervals and repeat the calculations for different values of "u" and "a" and tabulate the result.

Answer:

```
#include <iostream>
#include <malloc.h>
using namespace std;

class calculation{
        private:
                float initial_velocity;
                float time;
                float acceleration;
                float distance;
                float total_distance;

        public:
                void get(){
```

```
                cout<<"Please enter the following details to
calculate the distance travelled."<<endl;
                cout<<"Enter the initial velocity in m/s: ";
                cin>>initial_velocity;
                cout<<"Enter the time in seconds: ";
                cin>>time;
                cout<<"Enter the accleration in m/s^2: ";
                cin>>acceleration;
            }

            float distance_travelled(){
                float d;
                d = (initial_velocity*time) + (0.5* (acceleration
* time*time));

                distance = d;
                return d;
            }

            void display(){
                cout<<"The intial veloctiy is
"<<initial_velocity<<" m/s."<<endl;
                cout<<"The time taken is "<<time<<"
seconds."<<endl;
                cout<<"The acceleration is
"<<acceleration<<" m/s^2."<<endl;
                cout<<"The distance travelled is
"<<distance<<" meters."<<endl;
                cout<<"The total distance travelled till this
interval is "<<total_distance<<" meters."<<endl;

            }
            void insert_total_distance_travelled(float
total_distance_travelled){
                total_distance = total_distance_travelled;
            }

            float get_total_distance_travelled(){
                    return total_distance;
                }
        };

int main(){
        class calculation *interval;
        interval = (class calculation *) (malloc(sizeof(class
calculation)));
        float answer, total_d;
        static int interval_count = 0;
        int option;
        do{
                cout<<"\t\t\tThis program was made by Hari Krishna
Shah !!!"<<endl;
                cout<<"Welcome to the Main Menu."<<endl;
                cout<<"Choose an option to carry out an operation.\n\
                Enter 1 to perform a calculation.\n \
                Enter 2 to tabulate the result.\n \
                Enter 3 to quit the program."<<endl;
                cout<<"Enter you option here: ";
                cin>>option;

                switch(option){
                        case 1 :
                                interval = (class calculation *) (realloc(interval,
(interval_count + 1)*sizeof(class calculation)));
                                interval[interval_count].get();
                                answer =
interval[interval_count].distance_travelled();
                                cout<<"The distance travalled is
"<<answer<<" meters."<<endl;
                                if(interval_count == 0){
                                        total_d = answer;

        interval[interval_count].insert_total_distance_travelled(total_d);
                                }
                                else{
                                        total_d = answer + 
interval[interval_count-1].get_total_distance_travelled();

        interval[interval_count].insert_total_distance_travelled(total_d);
                                }
                                cout<<"The total distance travelled till this 
interval is "<<total_d<<" meters.\n"<<endl;
                                interval_count += 1;
                                break;
                        case 2:
                                if(interval_count == 0){
                                        cout<<"The calculations are empty. Do 
some calculations first.\n"<<endl;
                                }
                                else{
                                        for(int i = 0; i<interval_count; i++){
                                        cout<<"The details of the interval 
number "<<i+1<<" is given below."<<endl;
                                        interval[i].display();
                                        cout<<endl;
                                }
                                }
                                break;
                        case 3:
                                break;
                        default:
                                cout<<"Enter a valid option and try 
again.\n"<<endl;

                }
                cout<<endl;
        }
        while(option != 3);

        delete interval;
        cout<<"This program was made with love by Hari Krishna 
Shah.\nPlease drop a review or a feedback.\nThank You
```

```
!"<<endl;
        return 0;
}
```



C:\Users\Hari\OneDrive\Desktop\Object Oriented Programming\Theory Assignment\Assignment 1\ques 1.cpp - [Executing] - Dev-C++ 5.11

```
/*Ques 1. Construct an OOP to find the dist
seconds is given by
 Distance = u x t = ( (a x t2) / 2 ).
Here, “u” is the initial velocity (meters p
acceleration (meters per second2). Write ar
distance travelled at regular intervals of
and “a”. The program should provide the fle
their own time intervals and repeat the cal
values of “u” and “a” and tabulate the resu
*/


#include <iostream>
#include <malloc.h>
using namespace std;

class calculation{
        private:
            float initial_velocity;
            float time;
            float acceleration;
            float distance;
            float total_distance;

        public:
            void get(){
                cout<<"Please enter the following details to calculate the distance travelled."<<endl;
                cout<<"Enter the initial velocity in m/s: ";
                cin>>initial_velocity;
```

C:\Users\Hari\OneDrive\Desktop\Object Oriented Programming\Theory Assignment\Assignment 1\ques 1.exe

```
                        This program was made by Hari Krishna Shah !!!
Welcome to the Main Menu.
Choose an option to carry out an operation.
                Enter 1 to perform a calculation.
                Enter 2 to tabulate the result.
                Enter 3 to quit the program.
Enter you option here: 1
Please enter the following details to calculate the distance travelled.
Enter the initial velocity in m/s: 4
Enter the time in seconds: 5
Enter the accleration in m/s^2: 6
The distance travalled is 95 meters.
The total distance travelled till this interval is 95 meters.

                        This program was made by Hari Krishna Shah !!!
Welcome to the Main Menu.
Choose an option to carry out an operation.
                Enter 1 to perform a calculation.
                Enter 2 to tabulate the result.
                Enter 3 to quit the program.
Enter you option here: 1
Please enter the following details to calculate the distance travelled.
Enter the initial velocity in m/s: 1
Enter the time in seconds: 2
Enter the accleration in m/s^2: 4
The distance travalled is 10 meters.
The total distance travelled till this interval is 105 meters.

                        This program was made by Hari Krishna Shah !!!
Welcome to the Main Menu.
```

```
- Compilation Time: 1.06s
```

File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
Project Class ques 1.cpp
Compiler Resources Compile Log Debug Find Results Close
Abort Compilation
Line: 18 Col: 17 Sel: 0 Lines: 118 Length: 3750 Insert Done parsing in 0.484 seconds
10:12 PM 2022-03-18

Ques 2. At the end of every semester, a Faculty wants to prepare the Grade for the all student based on their scores in Internal Assessment Test1, Internal Assessment Test-2 and Final Assessment Test. These test marks are scored out of 100 and stored as Mark1, Mark2 and Mark3 against the respective student RegNo, Name and Government Identity Number (like Aathar No, Driving License No, Passport No etc…) respectively. Every student’s grade will be calculated based on their Page 2 of 2 average score in all the three marks. If the average score is >=90.0, the grade is ‘S’ and if the average score is >=80.0, the grade is ‘A’. Similarly, if the score is >=70.0, 60.0 and 50.0, the grade is assigned as ‘B’, ‘C’ and ‘D’ respectively. Otherwise, Grade is marked as ‘F’. Develop an object oriented program using classes and objects. Include the union to hold any one government Id card.

Answer:

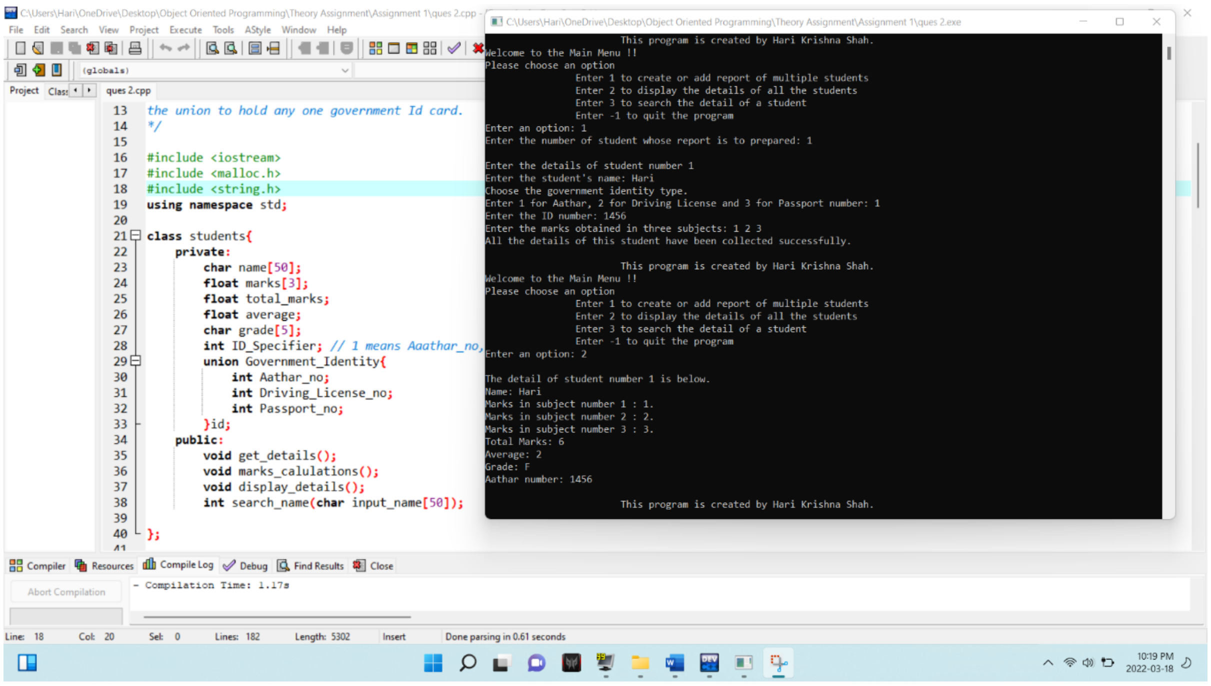
```
#include <iostream>
#include <malloc.h>
#include <string.h>
using namespace std;

class students{
	private:
		char name[50];
		float marks[3];
		float total_marks;
		float average;
		char grade[5];
		int ID_Specifier; // 1 means Aaathar_no, 2 means
Driving_License_no and 3 means Passport_no
		union Government_Identity{
			int Aathar_no;
			int Driving_License_no;
			int Passport_no;
		}id;
	public:
		void get_details();
		void marks_calulations();
		void display_details();
		int search_name(char input_name[50]);

};

void students::get_details(){
	cout<<"Enter the student's name: ";
	cin>>name;
	cout<<"Choose the government identity type.\nEnter 1 for
Aathar, 2 for Driving License and 3 for Passport number: ";
```

```
	cin>>ID_Specifier;
	cout<<"Enter the ID number: ";
	cin>>id.Aathar_no;
	cout<<"Enter the marks obtained in three subjects: ";
	for(int i = 0; i<3; i++){
		cin>>marks[i];
	}
	cout<<"All the details of this student have been collected 
successfully.\n"<<endl;

}

void students::marks_calulations(){
	float total = 0;
	for(int i = 0; i<3; i++){
		total += marks[i];
	}
	total_marks = total;
	average = total/3;
	if(average>=90){
		strcpy(grade, "S");
	}
	else if(average>=80){
		strcpy(grade, "A");
	}
	else if(average>=70){
		strcpy(grade, "B");
	}
	else if(average>=60){
		strcpy(grade, "C");
	}
	else if(average>=50){
		strcpy(grade, "D");
	}
	else{
		strcpy(grade, "F");
	}
```

```
}

void students::display_details(){
        cout<<"Name: "<<name<<endl;
        for(int i = 0; i<3; i++){
                cout<<"Marks in subject number "<<i+1<<" :
"<<marks[i]<<"."<<endl;
        }
        cout<<"Total Marks: "<<total_marks<<endl;
        cout<<"Average: "<<average<<endl;
        cout<<"Grade: "<<grade<<endl;
        if(ID_Specifier==1){
                cout<<"Aathar number: "<<id.Aathar_no<<endl;
        }
        if(ID_Specifier==2){
                cout<<"Driving License number:
"<<id.Aathar_no<<endl;
        }
        if(ID_Specifier==3){
                cout<<"Passport number: "<<id.Aathar_no<<endl;
        }
}
int students::search_name(char input_name[50]){
                        if(strcmp(input_name, name) ==0){
                                return 1;
                        }
                        else{
                                return 0;
                        }
}

int main(){
        class students *s;
        s = (class students *) (malloc(sizeof(class students)));
        static int student_count = 0;
        int temp1_count=0, temp2_count = 0;
        int option = 0;
        char name[50];

        while(option != -1){
                int search_result = 0;
                cout<<"\t\t\tThis program is created by Hari Krishna
Shah."<<endl;
                cout<<"Welcome to the Main Menu !!"<<endl;
                cout<<"Please choose an option\n \
                Enter 1 to create or add report of multiple students\n \
                Enter 2 to display the details of all the students\n \
                Enter 3 to search the detail of a student\n \
                Enter -1 to quit the program"<<endl;
                cout<<"Enter an option: ";
                cin>>option;
                switch(option){
                        case 1:
                                cout<<"Enter the number of student whose
report is to prepared: ";
                                cin>>temp2_count;
                                student_count += temp2_count;
                                cout<<endl;
                                s = (class students *) (realloc(s, student_count
* sizeof(class students)));
                                for(; temp1_count<student_count;
temp1_count++){
                                        cout<<"Enter the details of student
number "<<temp1_count+1<<endl;
                                        s[temp1_count].get_details();
                                        s[temp1_count].marks_calulations();
                                }
                                break;
                        case 2:
                                if(student_count == 0){
                                        cout<<"There isn't any student in the
class. Please add some students first"<<endl;
                                }
                                for(int i = 0; i<student_count; i++){
```

```
                    cout<<"\nThe detail of student number
"<<i+1<<" is below."<<endl;
                    s[i].display_details();
                }
                break;
            case 3:
                if(student_count == 0){
                    cout<<"There isn't any student in the
class. Please add some students first"<<endl;
                }
                else{
                    cout<<"Enter the name of the student
to search: ";
                    cin>>name;
                    cout<<endl;
                    for(int i = 0; i<student_count; i++){
                        search_result =
s[i].search_name(name);
                        if(search_result == 1){
                            s[i].display_details();
                        }
                    }
                    if(search_result ==0){
                        cout<<"The searched name was
not found. Please try again with another name."<<endl;
                    }
                }
                break;
            case -1:
                break;
            default:
                cout<<"Please enter a valid address and try
again."<<endl;
        }
        cout<<endl;

    }
```

```cpp
        delete s;
        cout<<"This program was made with love by Hari Krishna Shah.\nPlease drop a review or a feedback.\nThank You !"<<endl;
        return 0;
}
```



C:\Users\Hari\OneDrive\Desktop\Object Oriented Programming\Theory Assignment\Assignment 1\ques 2.cpp

```cpp
the union to hold any one government Id card.
*/

#include <iostream>
#include <malloc.h>
#include <string.h>
using namespace std;

class students{
    private:
        char name[50];
        float marks[3];
        float total_marks;
        float average;
        char grade[5];
        int ID_Specifier; // 1 means Aathar_no,
        union Government_Identity{
            int Aathar_no;
            int Driving_License_no;
            int Passport_no;
        }id;
    public:
        void get_details();
        void marks_calulations();
        void display_details();
        int search_name(char input_name[50]);

};
```

```
- Compilation Time: 1.17s
```

```
Line: 18    Col: 20    Sel: 0    Lines: 182    Length: 5302    Insert    Done parsing in 0.61 seconds
```

C:\Users\Hari\OneDrive\Desktop\Object Oriented Programming\Theory Assignment\Assignment 1\ques 2.exe

```
                        This program is created by Hari Krishna Shah.
Welcome to the Main Menu !!
Please choose an option
                Enter 1 to create or add report of multiple students
                Enter 2 to display the details of all the students
                Enter 3 to search the detail of a student
                Enter -1 to quit the program
Enter an option: 1
Enter the number of student whose report is to prepared: 1

Enter the details of student number 1
Enter the student's name: Hari
Choose the government identity type.
Enter 1 for Aathar, 2 for Driving License and 3 for Passport number: 1
Enter the ID number: 1456
Enter the marks obtained in three subjects: 1 2 3
All the details of this student have been collected successfully.

                        This program is created by Hari Krishna Shah.
Welcome to the Main Menu !!
Please choose an option
                Enter 1 to create or add report of multiple students
                Enter 2 to display the details of all the students
                Enter 3 to search the detail of a student
                Enter -1 to quit the program
Enter an option: 2

The detail of student number 1 is below.
Name: Hari
Marks in subject number 1 : 1.
Marks in subject number 2 : 2.
Marks in subject number 3 : 3.
Total Marks: 6
Average: 2
Grade: F
Aathar number: 1456

                        This program is created by Hari Krishna Shah.
```

File Edit Search View Project Execute Tools AStyle Window Help
(globals)
Project Class ques 2.cpp
Compiler Resources Compile Log Debug Find Results Close
Abort Compilation
10:19 PM
2022-03-18

3. i) Discuss any three advantages, disadvantages and applications of OOP.

ii) Differentiate the classes and objects.

iii) List out the various rules to define the constructors and destructors.

iv) Compare and contrast static binding and dynamic binding

v) When do we require static data members and static member functions? Give an example.

Answers:

3. i. Object-oriented programming (OOP) is a computer programming model that organizes software design around data, or objects, rather than functions and logic. An object can be defined as a data field that has unique attributes and behaviour.

OOP is a data centric approach which has features like data encapsulation, abstraction, inheritance and polymorphism. C++, Java, Ruby, Scala etc are some of the popular OOP languages.

Below are some of the advantages of object-oriented programming:

1. Re-usability

It means reusing some facilities rather than building them again and again. This is done with the use of a class. We can use it ‘n’ number of times as per our need.

2. Data Redundancy

This is a condition created at the place of data storage (you can say Databases) where the same piece of data is held in two separate places. So the data redundancy is one of the greatest advantages of OOP. If a user wants a similar functionality in multiple classes, he/she can go ahead by writing common class definitions for similar functionalities and inherit them.

3. Code Maintenance

This feature is more of a necessity for any programming languages; it helps users from doing re-work in many ways. It is always easy and time-saving to maintain and modify the existing codes by incorporating new changes into them.

4. Security

With the use of data hiding and abstraction mechanism, we are filtering out limited data to exposure, which means we are maintaining security and providing necessary data to view.

9. Polymorphism Flexibility

Let’s see a scenario to better explain this behavior.

You behave in a different way if the place or surrounding gets change. A person will behave like a customer if he is in a market, the same person will behave like a student if he is in a school and as a son/daughter if put in a house. Here we can see that the same person showing different behavior every time the surroundings are changed. This means polymorphism is flexible and helps developers in a number of ways.

i. It’s simplicity
ii. Extensibility

Below are some of the disadvantages of object-oriented programming:

1. **Steep learning curve**: The thought process involved in object-oriented programming may not be natural for some people, and it can take time to get used to it. It is complex to create programs based on interaction of objects. Some of the key programming techniques, such as inheritance and polymorphism, can be challenging to comprehend initially.
2. **Larger program size**: Object-oriented programs typically involve more lines of code than procedural programs.
3. **Slower programs**: Object-oriented programs are typically slower than procedure-based programs, as they typically require more instructions to be executed.

4. **Not suitable for all types of problems**: There are problems that lend themselves well to functional-programming style, logic-programming style, or procedure-based programming style, and applying object-oriented programming in those situations will not result in efficient programs.

Below are some of the applications of object-oriented programming:

1. Object-Oriented Databases

These databases try to maintain a direct correspondence between the real-world and database objects in order to let the object retain its identity and integrity. They can then be identified and operated upon.

2. Real-Time System Design

Real-time systems inherent complexities that make it difficult to build them. Object-oriented techniques make it easier to handle those complexities. These techniques present ways of dealing with these complexities by providing an integrated framework, which includes schedulability analysis and behavioral specifications.

3. Simulation and Modeling System

It's difficult to model complex systems due to the varying specification of variables. These are prevalent in medicine and in other areas of natural science, such as ecology, zoology, and agronomic systems. Simulating complex systems requires modeling and understanding interactions explicitly. Object-oriented programming provides an alternative approach for simplifying these complex modelling systems.

3. ii. Class: A class in C++ is the building block that leads to Object-Oriented programming. It is a user-defined data type, which holds its own data members and member functions, which can be accessed and used by creating an instance of that class. A C++ class is like a blueprint for an object.

Basically, it is a model of an entity.

Whereas, an object is an instance of a Class. It is an entity which is based on the model i.e class.

The differences between classes and objects are given below:

Class	Object
Class is the blueprint of an object. It is used to declare and create objects.	Object is an instance of class.
No memory is allocated when a class is declared.	Memory is allocated as soon as an object is created.
A class is a group of similar objects.	Object is a real-world entity such as book, car, etc.
Class is a logical entity.	Object is a physical entity.
Class can only be declared once.	Object can be created many times as per requirement.
Example of class can be car.	Objects of the class car can be BMW, Mercedes, jaguar, etc.

3. iii. Constructor in C++ is a special member function of a class whose task is to initialize the object of the class. A destructor is also a member function of a class that is instantaneously called whenever an object is destroyed.

The rules for defining constructors are given below:
1.Define a constructor if a class has an invariant

2. A constructor should create a fully initialized object

3. If a constructor cannot construct a valid object, throw an exception

4. Ensure that a value type class has a default constructor

5. Prefer default constructors to be simple and non-throwing

6. Don’t define a default constructor that only initializes data members; use member initializers instead

7. By default, declare single-argument constructors explicit

8. Define and initialize member variables in the order of member declaration

9. Prefer in-class initializers to member initializers in constructors for constant initializers

10. Prefer initialization to assignment in constructors

11. Use a factory function if you need “virtual behavior” during initialization

12. Use delegating constructors to represent common actions for all constructors of a class

13. Use inheriting constructors to import constructors into a derived class that does not need further explicit initialization

Syntax of constructor:

```
class XYZ

{

   ....

   XYZ()

   {

      //code here

   }

};
```

The rules for defining destructor are given below:

1) Name should begin with tilde sign(~) and must match class name.
2) There cannot be more than one destructor in a class.
3) Unlike constructors that can have parameters, destructors do not

allow any parameter.
4) They do not have any return type, just like constructors.
5) When you do not specify any destructor in a class, compiler generates a default destructor and inserts it into your code.

3 iv. Binding refers to the process of converting identifiers (such as variable and performance names) into addresses. Binding is done for each variable and functions. For functions, it means that matching the call with the right function definition by the compiler. It takes place either at compile time or at runtime.

When compiler acknowledges all the information required to call a function or all the values of the variables during compile time, it is called "**static binding**". As all the required information are known before runtime, it increases the program efficiency, and it also enhances the speed of execution of a program.

Calling a function or assigning a value to a variable, at runtime is called "**Dynamic Binding**". Dynamic Binding can be associated with run time 'polymorphism' and 'inheritance' in OOP.

Dynamic Binding makes the execution of program flexible as it can be decided, what value should be assigned to the variable and which function should be called, at the time of program execution. However, as this information is provided at runtime, it makes the execution slower as compared to static Binding.

The differences between static binding and dynamic binding are given below:

BASIS FOR COMPARISON	**STATIC BINDING**	**DYNAMIC BINDING**

Event Occurrence	Events occur at compile time are "Static Binding".	Events occur at run time are "Dynamic Binding".
Information	All information needed to call a function is known at compile time.	All information need to call a function come to know at run time.
Advantage	Efficiency.	Flexibility.
Time	Fast execution.	Slow execution.
Alternate name	Early Binding.	Late Binding.
Example	Overloaded function call, overloaded operators.	Virtual function in C++, overridden methods in java.

3.v. A static data member(s) is the data member(s) that gets memory only once for the whole class, no matter how many object(s) of a class is created. All objects of its class share this common copy.

Syntax:

```
class A
{
public:
   A() { cout << "A's Constructor Called " << endl;  }
};

class B
{
   static A a;
public:
   B() { cout << "B's Constructor Called " << endl; }
};
```

The Static Member Functions are those which are declared by using the Static in Front of the Member Function. It is possible to have static member functions in a class in the same way as static

data members. The static member function(s) is similar to the normal member function(s) of a class, but the only difference is that it can access only static member(s) (data or functions) declared in the same class. In other words, we cannot access non-static members inside the definition of a static member function.

```
#include <iostream>

using namespace std;

class Example{
   static int Number;
   int n;

   public:
   void set_n(){

       n = ++Number;
   }

   void show_n(){

       cout<<"value of n = "<<n<<endl;
   }

   static void show_Number(){

       cout<<"value of Number = "<<Number<<endl;
   }


};

int Example:: Number;
```