RACING REBELS

PLEASURE NEVER ENDS

AN OOP PROJECT

PROGRAMMERS:

ROLL NO	NAME
507	VISHWA DESAI
521	SAUMIL PANDYA
543	SOHAM VAGHELA
503	ANIKET BHARATI

CLASS: BE-II COMPUTER SCIENCE

SUBJECT LECTURER: MR. DIVYANG SIR

CONCEPT:

A popular type of gaming is a speed racing like f1 race and all others. Our game is also on this concept. Here you have to overtake your opponents and take care that you do this without fatal collision with your opponent cars. We have tried to include left and right turns, landscapes like mountain, clouds, a racing track and sign boards. The most attracting thing of our game is a racing car. It is nice model designed using c++ graphics. So overall we have tried to include the entire options available in such kind of racing games at our level.

Hope you will enjoy it to the fullest..

HEADER FILES

- ✓ #INCLUDE<STDLIB.H>

IT'S MEMBERS CONTAINS DIFFERENT CATEGORIES SUCH AS MEMORY CONTROL AND PROCESS CONTROL.

- ✓ #INCLUDE<IOSTREAM.H>

IT INCLUDES INPUT AND OUTPUT STREAM FOR VARIOUS INPUT AND OUTPUT OPERATIONS. ALL C++ FILES MUST INCLUDES THIS CLASS.

- ✓ #INCLUDE<STDIO.H>

IT IS STANDARD INPUT/OUTPUT FILES WHICH CONTAINS DECLARATIONS OF VARIOUS FUNCTIONS.

- ✓ #INCLUDE<FSTREAM.H>

IT IS A HEADER FILE REQUIRED TO HANDLE FILE PARAMETERS IN C++.

- ✓ #INCLUDE<CONIO.H>

  IT IS A HEADER FILE USED IN MS-DOS COMPILERS TO CREATE TEXT UESR INTERFACES.

- ✓ #INCLUDE<STRING.H>
  IT IS THE HEADER IN THE C STANDARD LIBRARY FOR THE C PROGRAMMING LANGUAGE ,WHICH CONTAINS MACRO DEFINITIONS,CONSTANTS,AND DECLARATIONS OF FUNCTIONS .
- ❖ COUT& CIN ARE THE OBJECTS OSTREAM & ISTREAM CLASSES

- ✓ #INCLUDE<GRAPHICS.H>

IT IS A HEADER FILE WHICH IS USED FOR GRAPHICALLY FORMATTED DISPLAY OF DATA. IT USES BASIC FUNCTION LIKE INITGRAPH TO DETECT AND INITIALIZE GRAPHIC DRIVER AND GRAPHIC MODE.

This game includes <u>OOP CONCEPTS</u> summarised as follows:

A) Data abstraction and encapsulation:

- ✓ The wrapping up of data and functions into a single unit (called class) is known as encapsulation.
- ✓ Data encapsulation is the most striking feature of the class. Data is not accessible to outside world, and those functions which are wrapped inside a class can access it.
- ✓ These functions provide the inter face between the object's data the program. This insulation of data from direct access by the program is called "data hiding".
- ✓ Abstraction refers to the art of representing essential features without including the background details or explanations.
- ✓ Since the classes use the concept of data abstraction, they are known as abstract data types (ADT).

B) Constructor:

- ✓ Constructor is a special function which enables an object to initialize itself when it is created. It is also known as automatic initialization of an object

- ✓ Default constructor – It refers to a constructor that is automatically generated in the absence of explicit constructors. A constructor that can take no arguments

C) Copy constructor:

- ✓ A **copy constructor** is a special constructor in the c++ programming language used to create new object as a copy of an existing object
- ✓ The first argument of such a constructor is a reference to an object of the same type as is being constructed, which might be followed by parameters of any type (all having default values).

D) Message passing:

- ✓ An Object oriented programming consist of a set of objects that communicate with each other.
- ✓ The process of programming in an object object-oriented language involves following:
  - Creating classing that define objects and their behaviour
  - Creating objects from class definition
  - E

    stablishing communication among objects
- ✓ The concept of massage passing makes easier to program real world problems.

E) Friend Function:

- ✓ A friend function is used in object-oriented programming to allow access to private or protected data in a class from outside the class.
- ✓ Normally a function which is not a member of a class cannot access such information; neither can an external class. Occasionally such access will be advantageous for the programmer; under these circumstances, the function or external class can be declared as a friend of the class using the friend keyword. The function or external class will then have access to all information – public, private, or protected – within the class.

F) I/O operations with file:

- ✓ A file can is a collection of data stored in a particular area on the disk. Program can use files for input output operation.
  - Data transfer between console unit and the program.
  - Data transfer between the program and a disk file.
- ✓ Fstream class is used to create objects for working with file.

File mode parameters:

Ios::app	-	Append to end-of-file
Ios::ate	-	Go to end-of-file at opening
Ios::binary	-	Binary file
Ios::in	-	Open file for reading only
Ios::out	-	Open file for writing only
Ios::trunc	-	Delete the content of the file if it exists

Other functions:

eof ();	-	Encounter end of file.
seekg ();	-	Moves get pointer to a specified location.

- ✓ File.write((char*)&obj,sizeof(obj))

It‘s first parameter is a casting parameter which breaks whole objects in to characters & store them in a file. It’s second parameter is to allocate memory to store objects.

- ✓ File.read((char*)&obj,sizeof(obj))

Its first parameter is casting parameter which breaks whole char in the form of objects. It's second parameter is to allocate memory to store objects.

G) Inheritance:

- ✓ Object oriented programming strongly supports the concept of reusability. The c++ classes can be reused in several ways.
- ✓ Once a class has been written and tested, it can be adopted by other programmers to suit their requirements. This is basically done by creating new classes, reusing properties of existing once.
- ✓ The mechanism of deriving a new class from an old one is called inheritance. The old class is referred to as the base class and the new one is called the derived class.
- ✓ Single inheritance, multiple inheritance, hierarchical inheritance, multilevel inheritance and hybrid inheritance are the type of inheritance.
- ✓ When a base class privately inherited in derived class then all the member becomes private to that derived class. If inherited publically, then private members of base class remains private and public member remains public.

Graphics

Initgraph:

- ✓ **InitGraph** initializes the graph package. It has 3 parameters.
  1. Graphic driver
  2. Graphic mode
  3. Path to bgi files(needed to include graphics)
- ✓ GraphDriver has two valid values: GraphDriver =0 which performs an auto detect and initializes the highest possible mode with the most colours. 1024x768x64K is the highest possible resolution supported by the driver.
- ✓ For another mode, GraphDriver needed to be set to a value different from zero and graphmode to the any other mode.

GRAPHIC FUNCTIONS

1)Line: requires co-ordinates of two points and draws line in between them.

2)Rectangle: requires co-ordinates of two points (upper left & lower right) and draws rectangle considering the points diagonally.

3) Bar: requires co-ordinate of two diagonal points and draws bar.

4) Arc: require co-ordinates(x, y) of centre of arc, starting and end angle and radius of arc as 5 parameters respectively and draws arc.

5) Setcolor: Set the colour from available set.

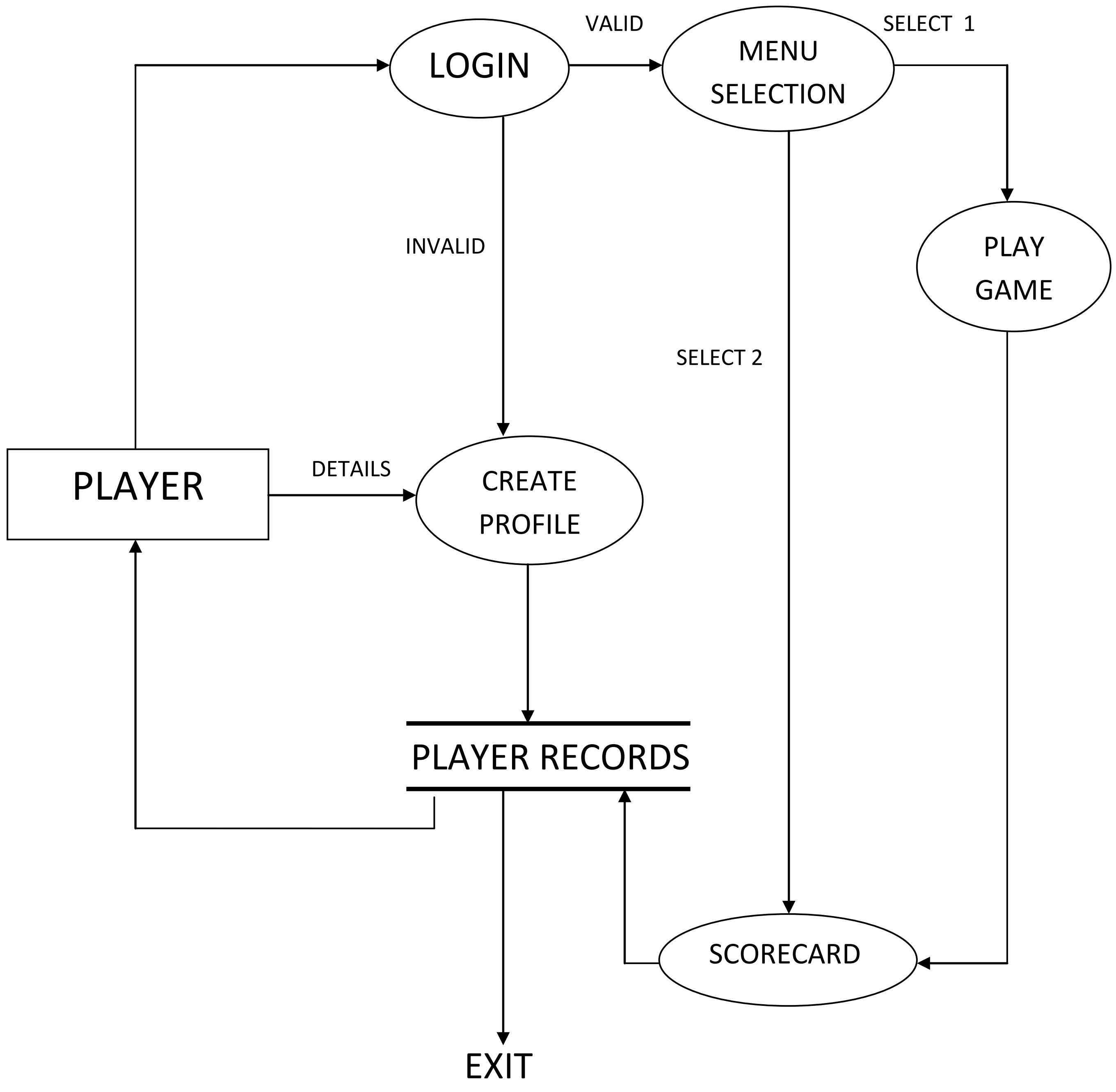
6) Setfillstyle: It sets the pattern and colour to fill by floodfill. It requires pattern no and colour as argument.

7) Floodfill: requires a co-ordinate of the point from where start filling the colour and the colour of line up to which the fill continues.

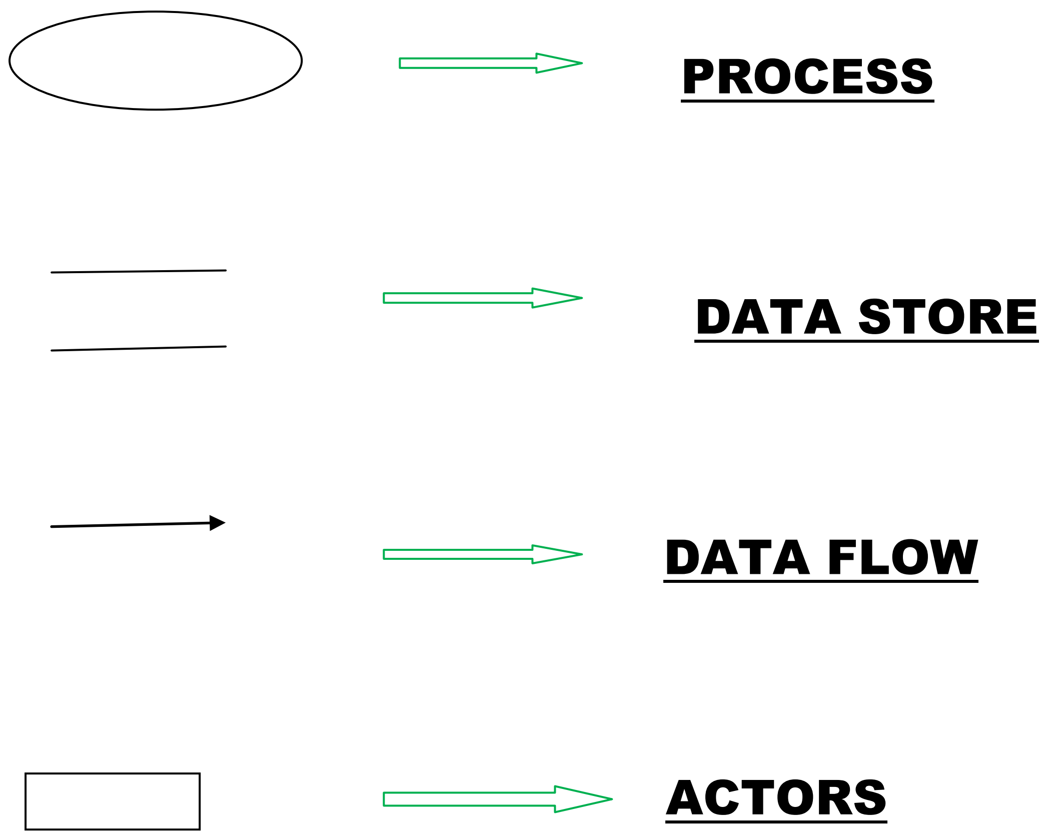
8) Settextstyle: This is a function for formatted text style. It requires text pattern, direction (0 for horizontal and 1 for vertical) and size of text as parameters.

9) Outtextxy: This is a function for formatted string display. It requires the string and co-ordinate of point from where display this string.

DATA FLOW DIAGRAM



<u>DATA FLOW DIAGRAM SYMBOLS</u>



CLASS DIAGRAM

Class obstacle

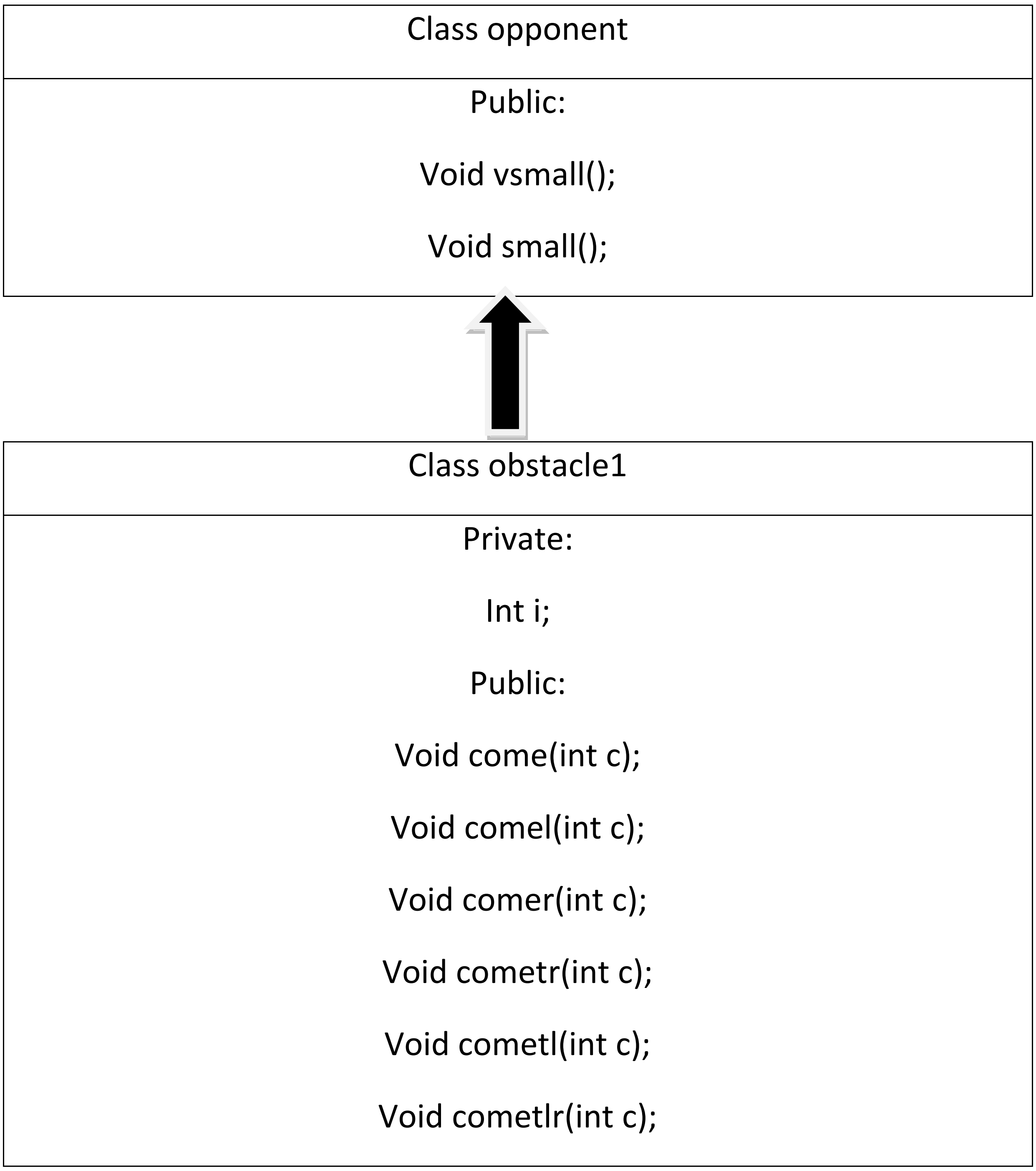
Private :
 	
Int i;
 	
Public:
 	
Void movel(int c);
 	
Void hordingl(int c);
 	
Void hordingr(int c);
 	
Void countdown1();
 	
Void countdown2();
 	
Void countdown3();
 	
Void movear_l(int c);

Class car
Public:

Int a,b;
Car();
Void design();
Void design1();

Class obstaclebox
Private:
Int i;
Int color;
Public:
Obstaclbox();
Void move(int c);
Void movel(int c);

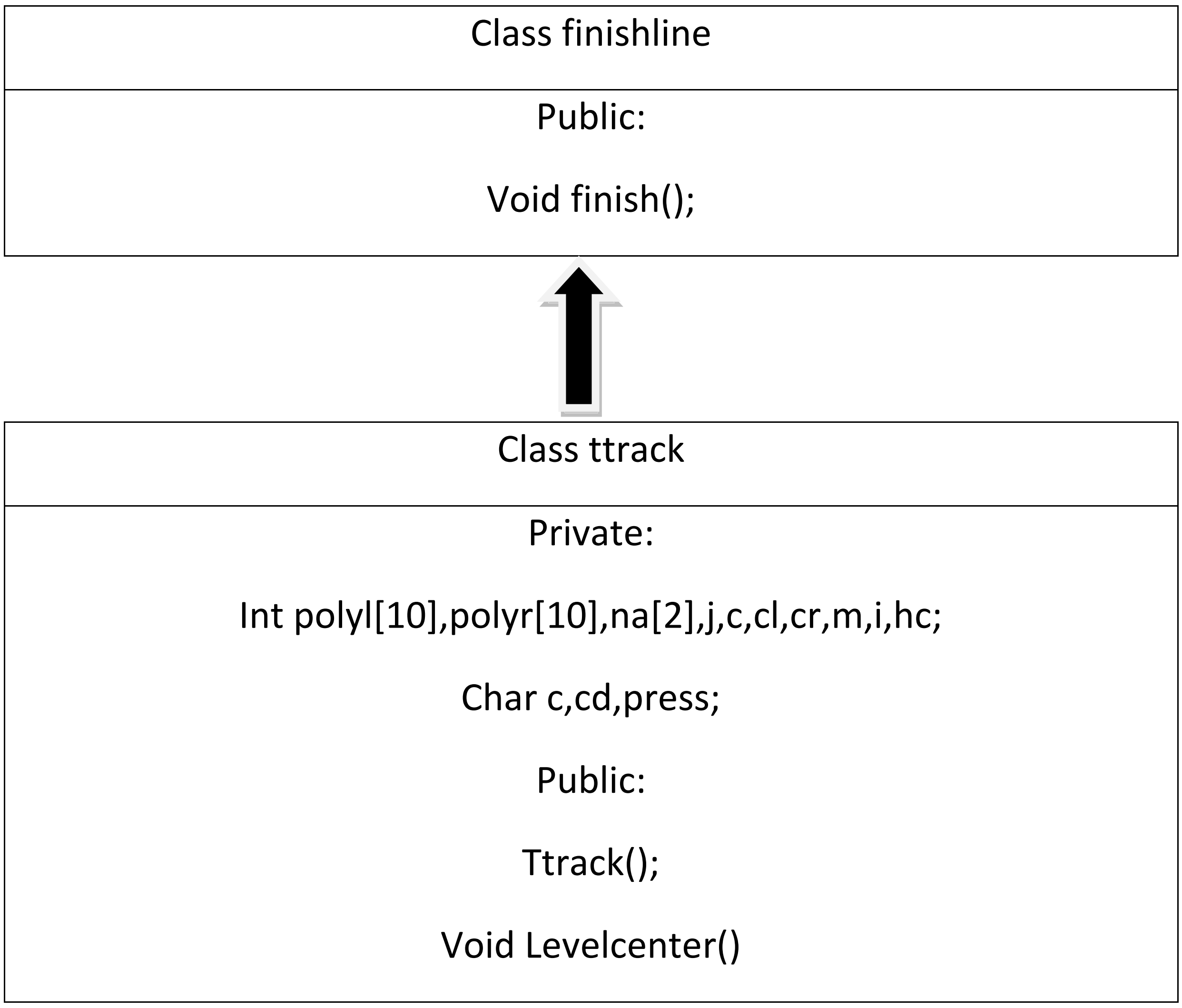
Class horizon
Private:
Int i,j;
Public:
Horizon();
Void cloud();



```
Class opponent
Public:
Void vsmall();
Void small();
```

```
Class obstacle1
Private:
Int i;
Public:
Void come(int c);
Void comel(int c);
Void comer(int c);
Void cometr(int c);
Void cometl(int c);
Void cometlr(int c);
```

Void cometrl(int c);



Class finishline

Public:

Void finish();

Class ttrack

Private:

Int polyl[10],polyr[10],na[2],j,c,cl,cr,m,i,hc;

Char c,cd,press;

Public:

Ttrack();

Void Levelcenter()

Class prof

Public:
 	
Char na[15],ti[9],no[2];
 	
Prof();
 	
Prof(prof &);
 	
Void input();
 	
Friend void write (prof &);
 	
Friend void read(int);
 	
Friend int compare(prof &);
 	
Friend int total();

LIMITATIONS

DUE TO THE LIMITATION OF THE COMPILER (64 KB RANGE) WE CAN NOT INCLUDE MORE NUMBER OF OBSTACLES AND OPPONENTS.

IT ALSO CREATES PROBLEMS WHEN TWO SIMULTANEOUS KEY STROKES ARE ENCOUNTERED.

ACKNOWLEDGEMENT

We wish to express our heartfelt appreciation to those who made their contribution to these projects and make this project work successfully and involve real life aspects.

We wish to express our profound gratitude to Prof.P.R.Bhavsar , Head of Comp & Engg Dept,Faculty of Technology and Engg,M.S.University of Baroda,for providing us this opportunity to take this project as part of our syllabus.

We extend our sincere thanks to our internal guide Divyang sir ,Lecturer of Dept of Comp & Engg for suggestions and support during the project period .

Last but not least we would like to thank our family members and friends for their support through out the duration of the project.

About The Project

This project uses the concept of data file handling and other concepts of "object oriented programming". Its Aim is to holds all aspect about

"A RACING GAME".