CLASS AND OBJECTS

**2008 Delhi:**

**2.a)** Differentiate between public and private visibility modes in context of Object Oriented Programming using a suitable example illustrating each.

**Ans: public and private visibility modes in context of OOP:**  
The visibility mode (private or public or protected) in the definition of the derived class specifies whether the features of the base class are privately derived or publicly derived or protected derived. The visibility modes basically control the access specifier to be for inheritable members of base class, in the derived class.

**Public visibility mode:** The public derivation means that the derived class can access the public and protected members of the base class but not the private members of the base class. With publicly derived class, the public members of the base class become the public members of the derived class, and the protected members of the base class become the protected members of the derived class.

**Private visibility mode:**The private derivation means, the derived class can access the public and private members of the base class privately. With privately derived class, the public and protected members of the base class become private members of the derived class. That means the inherited members can be accessed only through member functions of the derived class.

|  |  |  |
| --- | --- | --- |
| **Visibility Mode** | Inheritable public member becomes ( in derived class) | Inheritable protected member becomes (in derived class) |
| **public** | Public | protect |
| **private** | Private | private |

**public and private access specifiers in context of OOP:** public access specifier isused to define any method or a variable which may be accessed by any member function of the same class and also from outside the class. Private access specifier is used to make any variable or a method which has a limited access within the class only.The concept of private access specifier only.

**Eg:**

class student  
{ private:  
int rno;  
char name[21];  
public:  
int age;  
void input( );  
void display( );  
}

Here, since rno and name are declared in private, they can be accessed only inside the class. Since age,input( ) and display() are declared in public, they can be accessed from outside class also.

**2008 Outside Delhi:**

**2.a)** Differentiate between private and protected visibility modes in context of objectoriented programming using a suitable example illustrating each.

**Ans:private and protected visibility modes in context of OOP:**  
The visibility mode (private or public or protected) in the definition of the derived class specifies whether the features of the base class are privately derived or publicly derived or protected derived. The visibility modes basically control the access specifier to be for inheritable members of base class, in the derived class.

**Private visibility mode:**The private derivation means, the derived class can access the public and private members of the base class privately. With privately derived class, the public and protected members of the base class become private members of the derived class. That means the inherited members can be accessed only through member functions of the derived class.

**Protected visibility mode:** The protected derivation means that the derived class can access the public and private members of the base class protectedly. With protectedlyderived class, the public and protected members of the base calss become protected members of the derived class. That means the inherited members are now not available tothe outside world and can be accessed only through the member functions of the derived class and the classes based upon the derived classes. These members can be inherited further if any classes are inheriting from the derived class.

|  |  |  |
| --- | --- | --- |
| **Visibility Mode** | Inheritable public member becomes (inderived class) | Inheritableprotected member becomes (in derived |
| **protected** | Protected | Protected |
| **private** | Private | private |

**private and protected access specifiers in context of OOP:**  
private access specifier is used to make any variable or a method which has a limited access within the class only.At the time of inheritance, these variables cannot be accessed (inherited) to the derived class.protected access specifier is used to make any variable or a method which has a limitedaccess within the class only (here like private).But at the time of inheritance, these variables can be inherited to the derived class. Except regarding inheritance, both access specifiers ie private and protected will work same.

**Eg:**

class student  
{ private:  
int rno;  
char name[21];  
protected:  
int age;  
void input( );  
void display( );  
}

Here, since rno and name are declared in private, they can be accessed only inside the class. Since age,input( ) and display() are declared in protected, they also can be accessed only inside the class but they can be inherited, where as private members (rno and name) cannot be inherited.

**2006 Delhi:**

**2.c)** Define a class named ADMISSION in C++ with the following descriptions:

**Private Members**:

AD\_NO   integer(Ranges 10 – 2000)  
NAME     Array of characters(String)   
CLASS   Character   
FEES       Float

**Public Members:**

Function Read\_Data( ) to read an object of ADMISSION type. Function Display( ) to display the details of an object. Function Draw-Nos.( ) to choose 2 students randomly. And display the details. Use random function to generate admission nos. to match with AD\_NO.

**Ans**:

class ADMISSION  
{ int AD\_NO;  
char NAME[31];  
char CLASS;  
float FEES;  
public:  
void Read\_Data( )  
{ cout<<"\nEnter the Admission Number: ";  
cin>>AD\_NO;  
cout<<"\nEnter the Student Name: ";  
gets(NAME);  
cout<<"\nEnter the Class: ";  
cin>>CLASS;  
cout<<"\nEnter the Fees: ";cin>>FEES;  
}  
void Display()  
{ cout<<"\nThe Admission Number of the student: "<<AD\_NO;  
cout<<"\nThe name of the Student: “ <<NAME;  
cout<<"\nThe Class of the Student:” <<CLASS;  
cout<<"\nThe Fees of the Student: “ <<FEES;  
}  
void Draw\_Nos();  
};  
void ADMISSION::Draw\_Nos( )  
{

//Dear Students, a test for you. Complete this member function.  
}

**2006 Outside Delhi:**

**1.b)** Illustrate the use of Inline function in C++ with the help of an example.

2

**Ans:** **INLINE FUNCTIONS:**The inline functions are a C++ enhancement designed to speed up programs. The coding of normal functions and inline functions is similar except that inline functions definitions start with the keyword inline.

**The working of inline functions:**  
 After writing any program, it is first compiled to get an executable code, which consists of a set of machine language instructions. When this executable code is executed, the operating system loads these instructions into the computer’s memory, so that each instruction is stored in a specific memory location. Thus, each instruction has a particular memory address.

 After loading the executable program in the computer memory, these instructions are executed step by step. When a function call instruction is encountered, the program stores the memory address of the instruction immediately following the function call statement, loads the function being called into the memory, copies argument values, jumps to the memory location of the called function, executes the function code, stores the return value of the function, and then jumps back to the address of the instruction that was saved just before executing the called function.

 With inline code, the compiler replaces the function call statement with the function code itself (this process is called expansion) and then compiles the entire code. Thus, with inline functions, the compiler does not have to jump to another location to execute the function, and then jump back as the code of the called function is already available to the calling program.

Inline functions run a little faster than the normal functions as function calling overheads are saved, however there is a memory penalty. If 10 times an inlinefunction is called, there will be 10 copies of the function inserted into the code.

 A function can be declared inline by placing the keyword inline before it. An inline function definition should be placed above all the functions that call it. The functions should be inlined only when they are small. Since for large functions, they will become memory penalty. The inlining does not work for following situations:

**a**. For functions that return values and are having a loop or a switch or a goto.  
**b.** For functions not returning values, if a return statement exists.  
**c.**If functions contain static variables.  
**d**. If the function is recursive(a function that calls itself).

**Inlining and the member functions:**  
The member function of a class, if defined within the class definition, are inlined by default. Therefore, only very small member functions should be defined within the class definition.

The member functions defined outside the class definition can be made explicitly inline by placing the keyword inline before their definition.  
Inline functions are best for small functions that are called often.The compiler may even ignore your attempt to linline a function if it consists more than 50 lines of code.

**2. c)** Define a class named HOUSING in C++ with the following descriptions:

**Private Members:**

REG\_NO    integer(Ranges 10-1000)  
NAME          Array of characters(String)  
TYPE          Character  
COST          Float

**Public Members:**

**Function Read\_Data( )**to rread an object of HOUSING type.  
**Function Display( )** to display the details of an object.  
**Function Draw\_Nos( )** to choose and displaym the details of 2 houses selected randomly from an array of 10 objects of type HOUSING. Use random function to generate the registration nos. to match with REG\_NO from the array.

4

**Ans:**

class HOUSING  
{ int REG\_NO;  
char NAME[31];  
char TYPE;  
float COST;  
public:  
void Read\_Data( )  
{ cout<<"\nEnter the House Registration Number: ";  
cin>>REG\_NO;  
cout<<"\nEnter the House Name: ";  
gets(NAME);  
cout<<"\nEnter the House Type: ";cin>>TYPE;  
cout<<"\nEnter the House Cost: ";  
cin>>COST;  
}  
void Display()  
{ cout<<"\nThe Registration Number of the  
House: "<<REG\_NO;  
cout<<"\nThe name of the House: “ <<NAME;  
cout<<"\nThe Type of the House: "<<TYPE;  
cout<<"\nThe Cost of the House: "<<COST;  
}  
void Draw\_Nos();  
};  
void HOUSING::Draw\_Nos( )  
{ //Dear Students, a test for you. Completethis member function.  
}

**2004:**

**2.b)** Declare a class myfolder with the following specifications:

**Private members of the class:**  
Filenames    an array of strig of size[10][25] .(to represent all the names of files inside myfolder)  
Availspace    long. (to represent total number of bytes available in myfolder)  
Usedspace  long. (to represent total number of bytes used in myfolder)

**Public members of the class:**  
Newfileentry() : A function to accept values of Filenames, Availspace and Usedspace from user.  
Retavailspace(): A function that returns the value of total kilobytes available(1 kilobyte=1024 bytes)  
Showfiles(): A function that displays the names of all the files in myfolder

**Ans:**

class myfolder  
{ char Filenames[10][25];  
long Availspace;  
long Usedspace;  
public:  
void Newfileentry( )  
{ cout<<"\nEnter any 10 file names: ";  
for(int i=0;i<=9;i++)  
{cout<<"\nEnter the "<<i+1<<" file name: ";  
gets(Filenames[i]);  
}  
cout<<"\nEnter the Available Space (In Kilobytes): ";  
cin>>Availspace;  
cout<<"\nEnter the Used Space (In Kilobytes): ";  
cin>>Usedspace;  
}  
long RetavailSpace( )  
{ ret Availspace;  
}  
void Showfiles( )  
{ cout<<"\nThe names of the files inmyfolder object....";  
for(i=0;i<=9;i++)  
{ puts(Filenames[i]);  
cout<<endl;  
}  
}

**2002:**

**2.a)** What do you understand about a member function? How does a member function differ from an ordinary function?

**Ans:**A member function is a function declared within a class. It is said to be defined in two ways. Ie Outside the class and inside the class. When a member function is defined outside the class, the name of the function must be the full name including the class name as well. When a member function is defined inside the class, the name of the function is similar to an ordinary function but  
it will become an inline function.

**2.b)** Define a class Student for the following specifications.

**Private members of the Student are:**  
roll\_no          integer  
name            array of characters of size 20  
class\_st       array of characters of size 8  
marks           array of integers of size 5,   
Percentage  float  
**Calculate( )** that calculates overall percentage marks and returns the percentage

**Public Members of the Student are:**

**Readmarks** reads mark and invoke thecalculate function  
**Displaymarks** prints the data.

**Ans:**

class Student  
{ int roll\_no;  
char name[20];  
char class\_st[8];  
int marks[5];  
float percentage;  
float calculate( )  
{ percentage=(marks[0]+marks[1]+marks[2]+ marks[3]+marks[4])/5;  
return percentage;  
}  
public:  
void Readmarks( )  
{ cout<<”\nEnter any 5 subject marks;  
cin>>marks[0]>>marks[1]>>marks[2]>> marks[3]>>marks[4];  
calculate( );  
}  
void Displaymarks( )  
{ cout<<”\nThe Roll Number of the Student: “<<roll\_no;  
cout<<”\nThe Name of the Student:” <<name;  
cout<<”\nThe class of the Student: “ <<class\_st;  
cout<<”\n5 subject marks of the student…\n”;  
cout<<marks[0]<<”\t”<<marks[1]<<”\t”<<marks[2]<<”\t”;cout<<marks[3]<<”\t”<<marks[4]<<”\n”;  
cout<<”Percentage =”<<percentage;  
}  
};

**2001:**

**2.b)**Declare a class to represent bank account of 10 customers with the following data members. Name of the depositor, account number, type of account (S for Savings and C for Current), Balance amount. The class also contains member functions to do the following:

(i)To initialize data members.  
(ii) To deposit money  
(iii)To withdraw money after checking the balance (minimum balance is Rs.1000)  
(iv) To display the data members.

[Note:You are also required to give detailed function definitions.]

class Bank  
{ char name[15];  
int acc\_no;  
char acc\_type;  
float bal\_amount;  
public:  
void readData( )  
{ cout<<”\nEnter the name: “;  
gets(name);  
cout<<”\nEnter the account number: “;  
cin>>acc\_no;  
cout<<”\nEnter the account type: “;  
cin>>acc\_type;  
cout<<”\nEnter the amount to deposit: “;  
cin>>bal\_amount;  
}  
void deposit( )  
{ float deposit;  
cout<<”\nEnter your account number: “;  
cin>>acc\_no;  
cout<<”\nEnter the amount to deposit: “;  
cin>>deposit;  
bal\_amount=bal\_amount + deposit;  
}  
void withdraw( )  
{ float w\_amount;  
cout<<”\nEnter your account number:“;  
cin>>acc\_no;  
cout<<”\nEnter amount to withdraw”;  
cin>>w\_amount;  
if((bal\_amount-w\_amount)<1000)  
cout<<”\nWithdraw is not possible”;  
else  
{ bal\_amount=bal\_amount-w\_amount;  
cout<<”\nThe balance is“<<bal\_amount-w\_amount;  
}  
}  
void display( )  
{ cout<<”\nName of the depositor: “<<name;  
cout<<”\nAccount Number: “<<acc\_no;  
cout<<”\nAccount Type: “<<acc\_type;cout<<”\nThe balance amount is“<<bal\_amount;  
}  
};

**2000 :**

**2.b)** Define a class worker with the following specification.

**Private member of class worker:**  
wname            25characters  
hrwrk,wgrate   float (hours worked and wagerate per hour)  
totwage            float(hrwrk\*wgrate)  
cakcwg()          A function to find hrwrk\*wgrate with float return type

**Public members of class worker:**  
In\_data( ): A function to accept values forwno, wname, hrrwrk, wgrate and invoke  
calcwg( ) to calculate totpay.  
Out\_data( ): A function to display all the data members on the screen you should give definitions of functions.

class worker  
{ char wname[25];  
float hrwrk,wgrate;  
float totwage;  
float cakcwg( )  
{ return hrwrk\*wgrate;  
}  
public:  
void In\_data( )  
{ cout<<”\nEnter Worker number,name,hoursworked and wage rate”;  
cin>>wno;  
gets(wname);  
cin>>hrwrk>>wgrate;  
calcwg( );  
}  
void Out\_data( )  
{ cout<<”\nThe Worker Number: “<<wno;  
cout<<”\nThe Name of the worker: “<<wname;  
cout<<”\nNumber of hours worked by the worker: “<<hrwrk;  
cout<<”\nThe Wage Rate of the Worker: “<<wgrate;  
cout<<”\nThe total wages of the worker: “<<totwage;  
}

4

**1999 :**

**2.b)** Define a class Teacher with the following class specification:

**Private members:**

Name                  20 characters  
Subject               10 characters  
Basic, DA, HRA  float  
Salary                  float  
Calculate( ) function computes the salary and returns it. Salary is sum of Basic, DA and HRA

**Public members:**  
ReadData( ): Function accepts the data values and invoke the calculate function.  
DisplayData( ):Function prints the data on the screen.

class Teacher  
{ char Name[20];  
char subject[10];  
float Basic,DA,HRA,Salary;  
float Calculate( )  
{ Salary=Basic+DA+HRA;  
return Salary;  
}  
public:  
void ReadData( )  
{ cout<<"\nEnter Basic, Dearness Allowance and “;  
cout<<” House Rent Allowance: ";  
cin>>Basic>>DA>>HRA;  
Calculate();  
}  
void DisplayData( )  
{ cout<<"\nThe Basic : "<<Basic;  
cout<<"\nThe Dearness Allowance: "<<DA;  
cout<<"\nThe House Rent Allowance: "<<HRA;  
cout<<"\nThe Salary: "<<Salary;  
}  
};

**1998 Annual:**

**2.b)** Define a class student with the following specifications:

**Private members of class student:**

Admno   integer  
Sname   20 character  
English   float  
Math        float  
Science  float  
Total        float  
Ctotal( ) A function to  
calculate English + math + science with float return type

**Public member functions of class student:**  
Takedata( ):Function to accept values for admno,sname, English, math, science and invoke ctotal to calculate total.  
Showdata( ):Function to display all the data members on the screen.

class student  
{ int Admno;  
char Sname[20];  
float English,Math,Science,Total;  
float Ctotal()  
{ Total=English+math+science;  
return Total;  
}  
public:  
void Takedata()  
{ cout<<”\nEnter the admission number,name of the student: “;  
cin>>Admno;  
gets(sname);  
cout<<”\nEnter English, Maths, Science Marks: “;  
cin>>English>>Math>>Science;  
Ctotal( );  
}  
void Showdata( ){ cout<<”\nThe admission number of the student: “<<Admno;  
cout<<”\nThe name of the student: “<<Sname;  
cout<<”\nEnglish , Maths and Science Marks are…”;  
cout<<english<<”\t”<<math<<”\t” <<science<<”\n”;  
cout<<”\nTotal marks of the student: “<<Total;  
};