## **SECTION A**

### Question 1

Solution:

1. The differences between constructor and destructor are given below:

|  |  |
| --- | --- |
| CONSTRUCTORS | DESTRUCTORS |
| 1. Constructor is a special method used to initialize the instance(s) of a class. | 1. Destructor is used to destroy the objects after their scope has finished. |
| 1. Constructor is invoked whenever a new instance of a class is created. | 1. Destructor is invoked automatically when objects are destroyed. |
| 1. Constructor allocates the memory. | 1. Destructors are used to release the occupied memory. |
| 1. There are three types of constructors:    1. Default constructor    2. Parameterized constructor    3. Copy constructor | 1. Destructors don’t take arguments. |
| 1. Constructor overloading is possible. | 1. Destructor overloading is not possible. |
| 1. Syntax for constructor:   *Classname(arguments)*  *{*  *//body of constructor*  *}* | 1. Syntax for destructor:   *~ Classname ()*  *{*  *//body of destructor*  *}* |

1. The differences between call by value and call by reference are given below:

|  |  |
| --- | --- |
| CALL BY VALUE | CALL BY REFERENCE |
| 1. Syntax for call by value:    1. *returntype function\_name (type var1, type var2, …)*    2. *{*       1. *//body of function*    3. *}* | 1. Syntax for call by reference:   *returntype function\_name (type &var1, type &var2, …)*  *{*  *//body of function*  *}* |
| 1. In call by value, copies of the actual values are passed as argument. | 1. In call by reference, address of the original value is passed using pointer. |
| 1. The original values will not be affected if the values are modified in the function. | 1. The original values will also be modified if the values are modified in the function. |
| 1. Parameters passed as normal variables. | 1. Parameters passed are pointer variables. |

### Question 2

Solution:



int main()

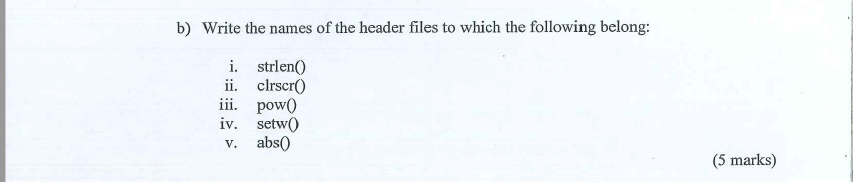
{

Seminar obj1; //for function 1

Seminar obj2(30) //for function 3

}

1. Function 4 is referred to as destructor. It is invoked when the objects’ scope has finished and they are destroyed. It releases the memory.
2. Function 1 and function 3 illustrate the concept of constructor overloading.



Solution:

1. strlen () – <string.h>
2. clrscr () – <conio.h>
3. pow () – <math.h>
4. setw () – <iomanip>
5. abs () – <stdlib.h>

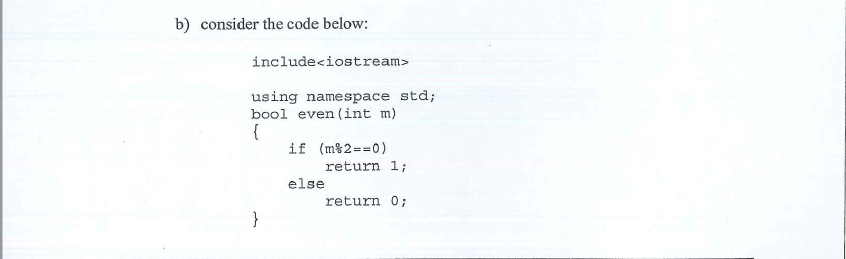
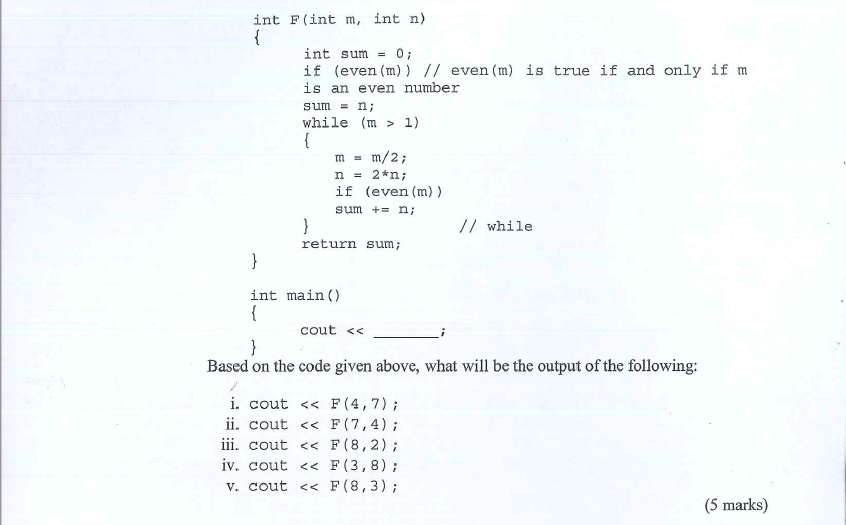
### Question 3

Solution:



Output:

33553

Solution:



Output:

21014021

### Question 4

Solution:



int testClass::sum()

{

return x+y;

}

void testClass::print() const

{

cout<<"x = "<<x<<endl;

cout<<"y = "<<y<<endl;

}



int main()

{

const testClass obj1;

const testClass obj2(10, 20);

obj1.sum();

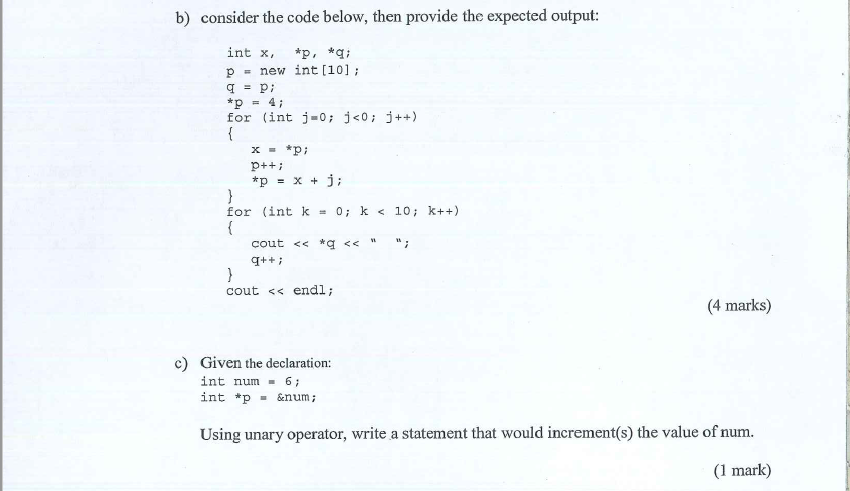
obj1.print();

obj2.sum();

obj2.print();

return 0;

}



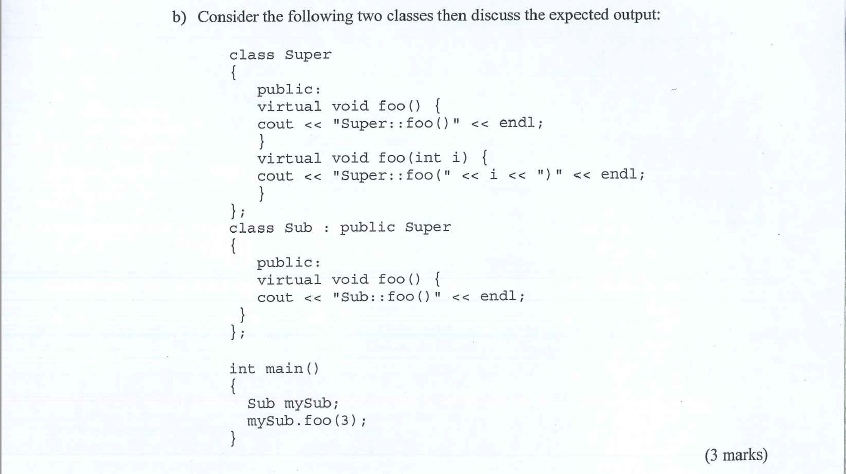


4 4 5 7 10 14 19 25 32 40



++(\*p);

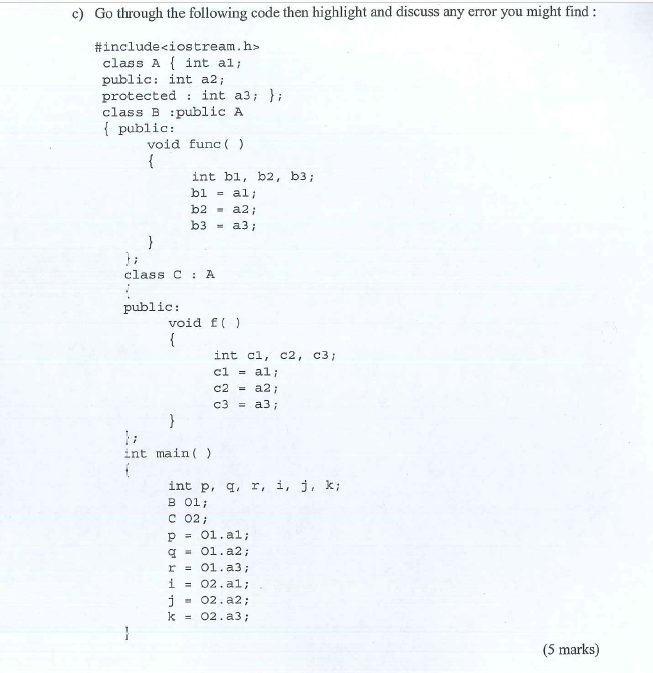
### Question 5

1. Solution:
2. Line1: #include<iostream.h>
3. Line 2: #include<stdio.h>
4. Line 8: public:
5. Line 16: E.Joining();
6. Solution:

Output:

A compile time error is received. It is because the function foo () with int type argument is declared as virtual in the base class Super. So, the child class Sub inheriting the virtual function should override the function to show the polymorphic behavior but overriding is not done.

So, compile time error is thrown by the given program.

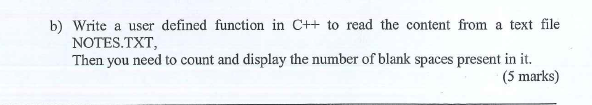
1.  Solution:
2. b1 = a1; in func () of class B is an error because a1 is private member variable of class A and is inaccessible to other classes except A itself.
3. Class C has inherited class A privately (by default), so the data members as well as member functions of A are not accessible to C. So, the entire function f () will show an error.
4. In main function, p=o1.a1; will be an error. Because o1 being object of class B, which inherits class A publicly, cannot access A’s private data member a1.
5. Again, in the main function, the statements from i=o2.a1; to k=o2.a3; will throw an error. Because o2 is an object of class C, which inherits class A privately. So, nothing in class A is accessible to class C.

## **SECTION B**

### Question 1

Solution:

1. Members accessible from member functions of class human are:
   1. iq\_level ()
   2. readmammal ()
   3. showmammal ()
   4. race [20]
   5. habitation [30]
2. Members accessed by an object of class mammal are the data members and member functions of class mammal only.
3. Members accessed by object of class human are:
   1. Data members and member functions of class human
   2. iq\_level ()
   3. readmammal ()
   4. showmammal ()
   5. race [20]
   6. habitation [30]



Solution:

#include<fstream>

#include<iostream>

using namespace std;

void readFile()

{

fstream fin;

fin.open(NOTES.txt, ios::in);

string word;

int count = 0;

int numspaces = 0;

while(!fin.eof())

{

fin>>word;

cout<<word;

count++;

}

cout<<"\n\nNumber of words in file are "<<count;

numspaces = count-1;

cout<<"\n\nNumber of blank spaces in file are "<<numspaces;

fin.close();

}

int main()

{

readFile();

return 0;

}

### Question 2

Solution:

Output:

x= 5/18

y= 5/18

z= 0/1

z= 5/18

w= 5/18

Press any key to close console window: c

Process returned 0 (0x0) execution time : 13.551 s

Press any key to continue.

### Question 3

Solution:

1. Constructor 1:

Sample::Sample()

{

this->x = 0;

this->y = 0.0;

}

1. Constructor 2

Sample::Sample(int val)

{

this->x = val;

this->y = 0.0;

}

1. Constructor 3

Sample::Sample(int val1, int val2)

{

this->x = val1;

this->y = val2;

}

Constructor 4

Sample::Sample(int val1, double val2)

{

this->x = val1;

this->y = val2;

}

## **SECTION C**

### Question 1

Solution:

#include<iostream>

using namespace std;

int main(){

try

{

throw 20;

}

catch(int n){

cout<<n<<" int type exception.\n";

}

try{

throw 'a';

}

catch(char n){

cout<<n<<" char type exception.\n";

}

}

### Question 2

Solution:

#include<iostream>

using namespace std;

template <class T>

void swapvalues(T &a, T &b)

{

T temp;

temp = a;

a = b;

b = temp;

}

int main()

{

int a, b;

cin>>a>>b;

cout<<"Before swap: "<<endl;

cout<<"A = "<<a<<", B = "<<b<<endl;

swapvalues(a, b);

cout<<"After swap: "<<endl;

cout<<"A = "<<a<<", B = "<<b<<endl;

char x, y;

cin>>x>>y;

cout<<"Before swap: "<<endl;

cout<<"X = "<<x<<", Y = "<<y<<endl;

swapvalues(x, y);

cout<<"After swap: "<<endl;

cout<<"X = "<<x<<", Y = "<<y<<endl;

return 0;

}

### Question 3

Solution:

#include<iostream>

using namespace std;

class Student

{

public:

char name[20];

int entryno;

Student(){ }

Student(int no)

{

entryno = no;

}

void getdata()

{

cout<<"Name : ";

cin>>name;

}

void display()

{

cout<<"Student Name : "<<name<<endl;

}

};

class Science: public Student

{

public:

Science()

{

cout<<"\nStream : Science"<<endl;

this->entryno = 1;

cout<<"Entry No : "<<entryno<<endl;

}

};

class Arts: public Student

{

public:

Arts()

{

cout<<"\nStream : Arts"<<endl;

this->entryno = 2;

cout<<"Entry No : "<<entryno<<endl;

}

};

int main()

{

Science o1;

o1.getdata();

o1.display();

Arts a1;

a1.getdata();

a1.display();

return 0;

}



#include<iostream>

using namespace std;

class Student

{

public:

char name[20];

int entryno;

Student(){ }

Student(int no)

{

entryno = no;

}

virtual void getdata()

{

cout<<"Name : ";

cin>>name;

}

virtual void display()

{

cout<<"Student Name : "<<name<<endl;

}

};

class Science: public Student

{

int phy, mat, che;

public:

Science()

{

cout<<"\nStream : Science"<<endl;

this->entryno = 1;

cout<<"Entry No : "<<entryno<<endl;

}

void getdata()

{

cout<<"\nEnter marks: "<<endl;

cout<<"Physics : "; cin>>phy;

cout<<"Mathematics : "; cin>>mat;

cout<<"Chemistry : "; cin>>che;

}

void display()

{

cout<<"\nSTUDENT'S MARKS"<<endl;

cout<<"Physics : "<<phy<<endl;

cout<<"Mathematics : "<<mat<<endl;

cout<<"Chemistry : "<<che<<endl;

}

};

class Arts: public Student

{

int eng, his, eco;

public:

Arts()

{

cout<<"\nStream : Arts"<<endl;

this->entryno = 2;

cout<<"Entry No : "<<entryno<<endl;

}

void getdata()

{

cout<<"\nEnter marks: "<<endl;

cout<<"English : "; cin>>eng;

cout<<"History : "; cin>>his;

cout<<"Economics : "; cin>>eco;

}

void display()

{

cout<<"\nSTUDENT'S MARKS"<<endl;

cout<<"English : "<<eng<<endl;

cout<<"History : "<<his<<endl;

cout<<"Economics : "<<eco<<endl;

}

};

int main()

{

Science o1;

o1.getdata();

o1.display();

Arts a1;

a1.getdata();

a1.display();

return 0;

}