Unit VI

6 File Handling and Design Patterns

6.1 : Concept of Stream

Q.1 Define the term - stream. ☞ [SPPU : June-22, Marks 2]

Ans. :

- Stream is basically a channel on which the data flow from sender to receiver.
- An input object that reads the stream of data from a file is called **input stream.**
- The output object that writes the stream of data to a file is called **output stream.**

6.2 : Byte Stream and Character Stream Classes

Q.2 Explain byte stream classes in detail. ☞ [SPPU ; June-22, Marks 3]

Ans. :

- The byte stream is used for inputting or outputting the bytes.
- There are two super classes in byte stream and those are **InputStream** and **OutputStream** from which most of the other classes are derived.
- These classes define several important methods such as **read()** and **write()**.
- The input and output stream classes are as shown in Fig. 6.2.1 and Fig. Q.2.1.

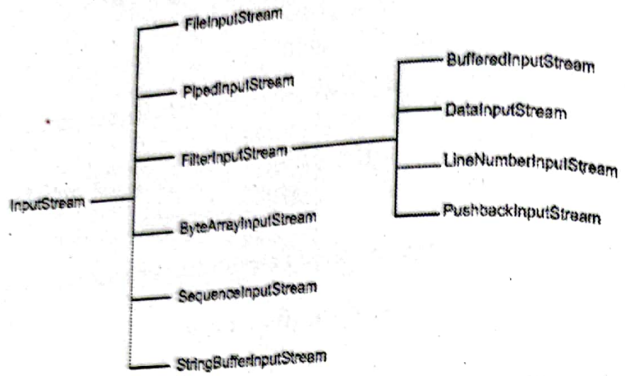


Fig. Q.2.1 InputStream classes

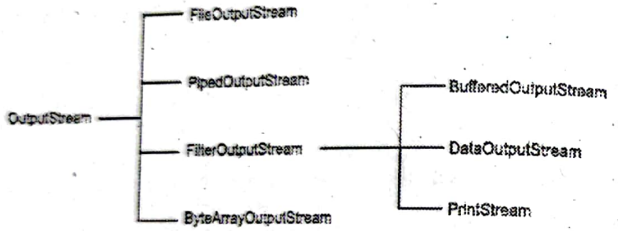


Fig. Q.2.2 OutputStream classes

Q.3 Explain character stream classes in detail.

☞ **[SPPU : June-22, Marks 4]**

Ans. :

- The character stream is used for inputting or outputting the characters.
- There are two superclasses in character stream and those are **Reader** and **Writer**.
- These classes handle the unicode character streams.
- The two important methods defined by the character stream classes are **read()** and **Write()** for reading and writing the characters of data.

- Various Reader and Writer classes are –

FileReader	FileWriter
PipeReader	PipeWriter
FilterReader	FilterWriter
BufferedReader	BufferedWriter
DataReader	DataWriter
LineNumberReader	LineNumberWriter
PushbackReader	PushbackWriter
ByteArrayReader	ByteArrayWriter
SequenceReader	SequenceWriter
StringBufferReader	StringBufferWriter

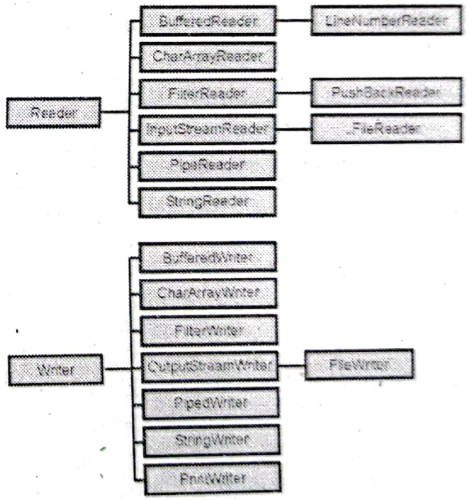


Fig. Q.3.1

6.3 : Using the File Class

Q.4 Write a constructor for file class.

Ans. :

- The file constructor is used in Java while handling the file I/O. These constructors are there in java.io.File. For example -

```
File f1=new File("input.dat");
```

Methods in class File :

1. **Public String getPath()** - This method returns a string that contains the path being used for the file.
2. **Public String getParent()** - This method returns a string that contains the name of the single directory which contains this file in the hierarchy.
3. **Public boolean exists()** - This method indicates whether or not a particular file exists where you expect it to be.
4. **Public boolean canWrite()** - This method indicates whether you have write access to this file.

Q.5 Write a Java program for obtaining the properties of file.

Ans. :

```
import java.io.File;
class FileProperties {
public static void main(String args[])
    {
        File FileObj=new File("test1/fun.txt");
        System.out.println("-----------------------------------------------");
        System.out.println("File Properties are as follows...");
        System.out.println("-----------------------------------------------");
        System.out.println("File Name: " + FileObj.getName());
        System.out.println("Path: " + FileObj.getPath());
        System.out.println("Abs Path: " + FileObj.getAbsolutePath());
        System.out.println("Parent: " + FileObj.getParent());
        System.out.println("This file exists: "+FileObj.exists());
```



```
        System.out.println(FileObj.canWrite() ?
        "is writeable" : "is not writeable");
        System.out.println(FileObj.canRead() ?
        "is readable" : "is not readable");
        System.out.println("is "+(FileObj.isDirectory() ? "" : " not "
        + " a directory"));
        System.out.println("It is a normal file: "+FileObj.isFile());
        System.out.println("This file is absolute:
        "+FileObj.isAbsolute());
        System.out.println("File last modified: " + new
        java.util.Date(FileObj.lastModified()));
        System.out.println("File size: " + FileObj.length() + " Bytes");
    }
}
```

6.4 : Input/Output Exceptions

Q.6 Describe the commonly used input and output exceptions.

Ans. :

I/O Exception Class	Purpose
IOException	If some sort of I/O related exception occurs then this class is used to handle it.
FileNotFoundException	This is the most commonly used exception during I/O operations using files. If the desired file for performing I/O is not found then this exception is used.
EOFException	If we come across end of file unexpectedly then this exception is raised.
InterruptedIOException	If I/O operation is interrupted , then this exception is used to handle the situation.

6.5 : Creation of Files

Q.7 How will you create a file using file stream class ? Explain with example.

Ans. :

- File stream object is an important entity which helps us to locate desired file using a filename. We need to create a file first and then initialize the file stream object.

Following sample code makes use **FileWriter** class

```
try{
FileWriter fwrt=new FileWriter("D:\\myfile.txt");
fwrt.write("Java Programming is Fun!!!");
fwrt.close();
}
catch(Exception e)
{
        System.out.println(e);
}
```

6.6 : Reading/Writing Character

Q.8 Write a program to copy contents of one file to another file using character stream class.

OR Write a java program to copy the content of the file "file1.txt" into new file"file2.txt". ☞ **[SPPU : June-22, Marks 8]**

Ans. :

```
import java.io.FileReader;
import java.io.FileWriter;
public class CopyFileProg
{
        public static void main(String args[])throws Exception{
                FileReader fr=new FileReader("D:\\file1.txt");
                FileWriter fwrt=new FileWriter("D:\\file2.txt");
                int i;
                while((i=fr.read())!=-1)
                {
```



```
                fwrt.write((char)i);
                }
                System.out.println("The Data is copied from file1 to file2");
                fr.close();
                fwrt.close();
        }
}
```

Output

```
The Data is copied from file1 to file2
D:\MSBTE_JAVA _Programs> cd..
D:\> type file1.txt
Java is fun!!!
D:\> type file2.txt
Java is fun!!!
D:\>
```

6.7 : Reading/Writing Bytes

Q.9 Write a program to copy contents of one file to another. Using byte stream classes.

Ans. :

```
import java.io.FileInputStream;
import java.io.FileOutputStream;
public class CopyFileProg1
{
        public static void main(String args[])throws Exception
        {
                FileInputStream fin=new FileInputStream("D:\\file1.txt");
                FileOutputStream fout=new
                FileOutputStream("D:\\file2.txt");
                int i;
                while((i=fin.read())!=-1)
                {
                        fout.write((char)i);
                }
                System.out.println("The Data is copied from file1 to file2");
                fin.close();
                fout.close();
        }
}
```

6.8 : Handling Primitive Data Types

Q.10 Write a Java program to write some data to a sample file and then read this sample file. Display the contents that you have read from the sample file.

Ans. :

```
import java.io.*;
class PrimitiveDemo
{
    public static void main(String args[]) throws IOException
    {
        //creating file and data stream object and
        initializing them for writing purpose
        File f=new File("d:\\Primfile.txt");
        FileOutputStream fos=new FileOutputStream(f);
        DataOutputStream dos=new DataOutputStream(fos);
        //Writing primitive data to the file
        System.out.println("\tWriting some data to the file ...");
        dos.writeInt(301);
        dos.writeChar('A');
        dos.writeBoolean(true);
        dos.writeDouble(123.45);
        //closing the output streams of
        filestream and datastream classes
        dos.close();
        fos.close();
        //initializing file and data stream
        objects for reading purpose
        FileInputStream fis=new FileInputStream(f);
        DataInputStream dis=new DataInputStream(fis);
        //Reading primitive data from the file
        System.out.println("\tReading some data from the file ...");
        System.out.println(dis.readInt());
        System.out.println(dis.readChar());
        System.out.println(dis.readBoolean());
        System.out.println(dis.readDouble());
```



```
        //closing the input streams of filestream
        and datastream classes
        dos.close();
        fos.close();
    }
}
```

6.9 : Concatenating and Buffer Files

Q.11 Write short note on – Concatenating and buffer files.

Ans. : Concatenation : This is the operation by which two or more files are saved in one file.

Buffer Files : In Java we can create a buffer to store temporary data that is read from and written to a stream and this process is known as input and output buffer operation.

Following Java program shows the concatenation of two files and displaying the result on Console.

```
import java.io.*;
class  ConcatAndBuffer
{
  public static void main(String[] args)
  {
      try{
        FileInputStream f1 = new FileInputStream("d:\\File1.txt");
        FileInputStream f2 = new FileInputStream("d:\\File2.txt");
        SequenceInputStream resultFile = null;

        resultFile = new SequenceInputStream(f1,f2);

        BufferedInputStream inBuffer =
            new BufferedInputStream(resultFile);

        BufferedOutputStream outBuffer =
            new BufferedOutputStream(System.out);
```

```
        int ch;
        while((ch = inBuffer.read())!=1){
            outBuffer.write((char)ch);
        }
        inBuffer.close();
        outBuffer.close();
        f1.close();
        f2.close();
      }catch(Exception e) {}
   }
}
```

6.10 : Random Access Files

Q.12 What is random access file ?

Ans. :

- The **RandomAccessFile** class in the Java enables us to read or write to a specific location in the file at the file pointer.
- The **java.io.RandomAccessFile.seek(long pos)** method sets the filepointer offset, which is set to the beginning of this file, at which the next read or write occurs. The pos is the offset position, measured in bytes from the beginning of the file. This method does not return a value. The IOException is used while using this method.

Java Program[RandomAccessFileDemo.java]

```
import java.io.*;
public class RandomAccessFileDemo
{
  public static void main(String[] args)
  {
    try {
      RandomAccessFile raf = new
      RandomAccessFile("d://names.dat","rw");
      String names[] = new String[5];
```



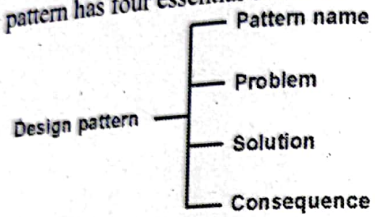
```
      names[0]  = "Archana";
      names[1]  = "Shilpa";
      names[2]  = "Sujata";
      names[3]  = "Soumya";
      names[4]  = "Shivraj";

      for (int i = 0; i < names.length; i++)
        {
        raf.writeUTF(names[i]);
      }
      raf.seek(raf.length());
      raf.writeUTF("Anuja");
      raf.seek(0);
      while (raf.getFilePointer() < raf.length())
        {
        System.out.println(raf.readUTF());
      }
       raf.seek(0);
     raf.writeUTF("Supriya");
       raf.seek(0);
       System.out.println("\t List After Modification...");
      while (raf.getFilePointer() < raf.length())
        {
        System.out.println(raf.readUTF());
      }
    }
    catch (FileNotFoundException e)
    {
      e.printStackTrace();
    }
    catch (IOException e)
    {
      e.printStackTrace();
    }

  }
}
```

6.11 : Introduction to Design Pattern

Q.13 What is design pattern ?

Ans. : Definition of design pattern : Design pattern is general reusable solution to commonly occurring problem within a given context in software design.

In general the pattern has four essential elements -



6.12 : Types of Design Patterns

Q.14 Explain the three types of design patterns.

Ans. :

(1) Creational Pattern

Creational design pattern are the design pattern that are used for object creation mechanism. This type of pattern is used in the situation when basic form of object creation could result in design problems or increase complexity of a code base.

The creational patterns show how to make the software design flexible.

There are well known design patterns that are part of creational pattern.

These are - Abstract factory, factory method, singleton pattern.

(2) Structural Pattern

- In software engineering, structural design patterns are design patterns that ease the design by identifying a simple way to realize relationships between entities.

- The structural design pattern serves as a blueprint for how different classes and objects are combined to form larger structures.
- Structural patterns are just similar to data structures.
- For example - Bridge, adapter, composite.

(3) Behavioral Pattern

- The behavioral design patterns are design patterns that identify common communication patterns between objects and realize these patterns. By doing so, these patterns increase flexibility in carrying out this communication.
- A behavioral pattern explains how objects interact.
- It describes how different objects and classes send messages to each other to make things happen and how the various tasks are divided among different objects.

6.13 : Adapter

Q.15 What is adapter pattern ? ☞ [SPPU : June-22, Marks 4]

Ans. :

Intent

- The adapter pattern allows the interface of existing class to be used from another interface.
- It is often used to make existing classes work with other without modifying their source code.
- The adapter helps two incompatible interfaces to work together.
- When one class relies upon a specific interface that is not implemented by another class, the adapter acts as a translator between the two types. Thus this pattern translates one interface for a class into another compatible interface.

Structure

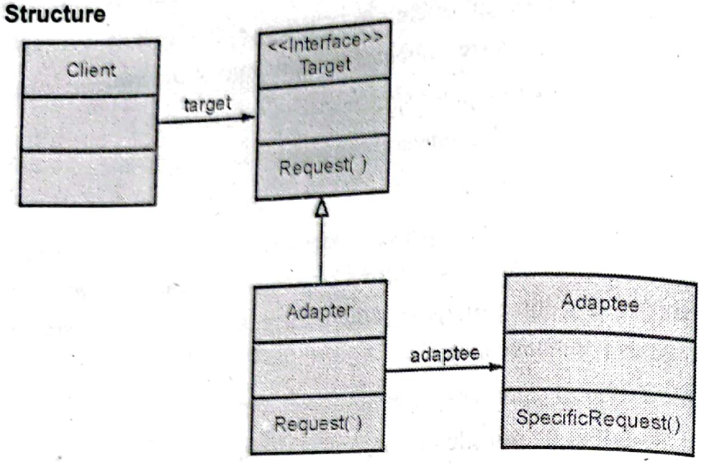


Fig. Q.15.1 : Representation

Application

Adapter pattern acts as a bridge between two compatible functionalities. The **bar code reader system** is a system in which the adapter pattern can be observed.

6.14 : Singleton

Q.16 What is singleton pattern ?

☞ **[SPPU : June-22, Marks 4]**

Ans. :

Intent

- The class has only one instance and it provides global point of access to it.

Motivation

- Singleton is a simplest pattern. This type of pattern belongs to creational design pattern.

- Singleton pattern is created in a situation in which **only one instance of a class** must be created and this instance can be used as a global point of access.
- Singleton work by having special method to get the instance of the desired object

Structure

Following diagram represents how to implement Singleton design pattern.

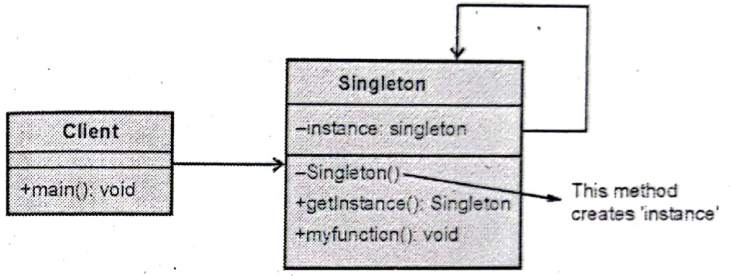


Fig. Q.16.1 : Representation of singleton

Application

Incometax Calculator is a typical example of singleton pattern. The client instantiates the instance for Incometax Calculator only once when needed.

6.15 : Iterator

Q.17 What is iterator pattern ? Give the merits and demerits of it.

OR What is adapter pattern ?

Ans. : The Iterator provides ways to access elements of an aggregate object sequentially without exposing the underlying structure of the object.

Merits

1. This design pattern accesses the contents of a collection without exposing its internal structure.

2. It supports multiple simultaneous traversals of a collection.
3. Using iterator uniform interface for traversing different collection is possible

Demerits

Iteration can be done only once. If you reach the end of the collection of elements, its done. If we need to iterate again we should get a new iterator.

Q.18 Description : Student management system using file handling. Use appropriate data members and methods.

i. **Create**

ii. **Display**

iii. **Search**

iv. **Modify**

v. **Delete** [SPPU : June-22, Marks 9]

Ans. : Implementation :

```
#include<iostream>
#include<iomanip>
#include<fstream>
#include<cstring>

using namespace std;
class Student_CLASS
{
        typedef struct STUDENT
        {
                char name[10];
                int Rollno;
        }Rec;
        Rec Records;
        public:
                void Create();
                void Display();
                void Update();
                void Delete();
                void Append();
```



```
                int Search();
};
void Student_CLASS::Create()
{
        char ch='y';
        fstream seqfile;
        seqfile.open("D:\\STUD.DAT",ios::in|ios::out|ios::binary);
        do
        {
                cout<<"\n Enter Name: ";
                cin>>Records.name;
                cout<<"\n Enter Rollno: ";
                cin>>Records.Rollno;
                //then write the record containing this data in the file
                seqfile.write((char*)&Records,sizeof(Records));
                cout<<"\nDo you want to add more records?";
                cin>>ch;
        }while(ch=='y');
        seqfile.close();
}
void Student_CLASS::Display()
{
        fstream seqfile;
        int n,m,i;
        seqfile.open("D:\\STUD.DAT",ios::in|ios::out|ios::binary);
        //positioning the pointer in the file at the beginning
        seqfile.seekg(0,ios::beg);
        cout<<"\n The Contents of file are ..."<<endl;
        //read the records sequentially
        while(seqfile.read((char *)&Records,sizeof(Records)))
        {
                if(Records.Rollno!=-1)
                {
                        cout<<"\nName: "<<Records.name;
                        cout<<"\nRollno: "<<Records.Rollno;
                        cout<<"\n";          }
}
int last_rec=seqfile.tellg();//last record position
//formula for computing total number of objects in the file
n=last_rec/(sizeof(Rec));
```

```
seqfile.close();
}
void Student_CLASS::Update()
{
        int pos;
        cout<<"\n For updation,";
        fstream seqfile;
        seqfile.open("D:\\STUD.DAT",ios::in|ios::out|ios::binary);
        seqfile.seekg(0,ios::beg);
//obtaining the position of desired record in the file
        pos=Search();
        if(pos==-1)
        {
        cout<<"\n The record is not present in the file";
        return;
        }
//calculate the actual offset of the desired record in the file
        int offset=pos*sizeof(Rec);
        seqfile.seekp(offset);//seeking the desired record for modification
        cout<<"\n Enter the values for updation...";
        cout<<"\n Name: ";
        cin>>Records.name;
        cout<<"\n Rollno: ";
        cin>>Records.Rollno;
        seqfile.write((char*)&Records,sizeof(Records))<<flush;
        seqfile.seekg(0);
        seqfile.close();
        cout<<"\n The record is updated!!!";
}
void Student_CLASS::Delete()
{
        int id,pos;
        cout<<"\n For deletion,";
        fstream seqfile;
        seqfile.open("D:\\STUD.DAT",ios::in|ios::out|ios::binary);
        seqfile.seekg(0,ios::beg);//seeking for reading purpose
        pos=Search();//finding pos. for the record to be deleted
        if(pos==-1)
        {
        cout<<"\n The record is not present in the file";
```



```
        return;
        }
//calculate offset to locate the desired record in the file
        int offset=pos*sizeof(Rec);
        seqfile.seekp(offset);//seeking the desired record for deletion
        strcpy(Records.name,"");
        Records.Rollno=-1;
        seqfile.write((char*)&Records,sizeof(Records))<<flush;
        seqfile.seekg(0);
        seqfile.close();
        cout<<"\n The record is Deleted!!!";
}
void Student_CLASS::Append()
{
        fstream seqfile;
        seqfile.open("D:\\STUD.DAT",ios::ate|ios::in|ios::out|ios::binary);
        seqfile.seekg(0,ios::beg);
        int i=0;
        while(seqfile.read((char *)&Records,sizeof(Records)))
        {
                i++;//going through all the records
                        // for reaching at the end of the file
        }
        //instead of above while loop
        //we can also use seqfile.seekg(0,ios::end)
        //for reaching at the end of the file
        seqfile.clear();//turning off EOF flag
        cout<<"\n Enter the record for appending";
        cout<<"\nName: ";
        cin>>Records.name;
        cout<<"\nRollno: ";
        cin>>Records.Rollno;
        seqfile.write((char*)&Records,sizeof(Records));
        seqfile.seekg(0);//reposition to start(optional)
        seqfile.close();
        cout<<"\n The record is Appended!!!";
}
int Student_CLASS::Search()
{
        fstream seqfile;
```

```
    int id,pos;
    cout<<"\n Enter the Rollno for searching the record ";
    cin>>id;
    seqfile.open("D:\\STUD.DAT",ios::ate|ios::in|ios::out|ios::binary);
    seqfile.seekg(0,ios::beg);
    pos=-1;
    int i=0;
    while(seqfile.read((char *)&Records,sizeof(Records)))
    {
        if(id==Records.Rollno)
        {
            pos=i;
            break;
        }
        i++;
    }
    return pos;
}
int main()
{
    Student_CLASS List;
    char ans='y';
    int choice,key;
    do
    {
    cout<<"\n          Main Menu          "<<endl;
    cout<<"\n 1.Create";
    cout<<"\n 2.Display";
    cout<<"\n 3.Update";
    cout<<"\n 4.Delete";
    cout<<"\n 5.Append";
    cout<<"\n 6.Search";
    cout<<"\n 7.Exit";
    cout<<"\n Enter your choice ";
    cin>>choice;
    switch(choice)
    {
        case 1:List.Create();
               break;
        case 2:List.Display();
               break;
```



```
        case 3:List.Update();
               break;
        case 4:List.Delete();
               break;
        case 5:List.Append();
               break;
        case 6:key=List.Search();
               if(key<0)
                   cout<<"\n Record is not present in the file";
               else
                   cout<<"\n Record is present in the file";
               break;
        case 7:exit(0);
    }
    cout<<"\n\t Do you want to go back to Main Menu?";
    cin>>ans;
    }while(ans=='y');
    return 0;
}
```

Output

```
Main Menu
1.Create
2.Display
3.Update
4.Delete
5.Append
6.Search
7.Exit

Enter your choice 1
Enter Name: AAA
Enter Rollno: 10
Do you want to add more records?y
Enter Name: BBB
Enter Rollno: 20
Do you want to add more records?y
Enter Name: CCC
Enter Rollno: 30
Do you want to add more records?n
    Do you want to go back to Main Menu?y
        Main Menu
```

```
1.Create
2.Display
3.Update
4.Delete
5.Append
6.Search
7.Exit
Enter your choice 2
The Contents of file are ...
Name: AAA
Rollno: 10

Name: BBB
Rollno: 20

Name: CCC
Rollno: 30
      Do you want to go back to Main Menu?y
        Main Menu
1.Create
2.Display
3.Update
4.Delete
5.Append
6.Search
7.Exit
Enter your choice 3
For updation,
Enter the Rollno for searching the record 20

Enter the values for updation...
Name: XXX
Rollno: 11

The record is updated!!!
      Do you want to go back to Main Menu?y
        Main Menu
1.Create
2.Display
3.Update
```



```
4.Delete
5.Append
6.Search
7.Exit
Enter your choice 2
The Contents of file are ...
Name: AAA
Rollno: 10

Name: XXX
Rollno: 11

Name: CCC
Rollno: 30

      Do you want to go back to Main Menu?n
```

END...

SOLVED MODEL QUESTION PAPER (In Sem)
Object Oriented Programming
S.E. (IT) Semester - III [As Per 2019 Pattern]

Time : 1 Hour] **[Maximum Marks : 30**

N.B.

i) Attempt Q.1 or Q.2, Q.3 or Q.4.

ii) Neat diagrams must be drawn wherever necessary.

iii) Figures to the right side indicate full marks.

iv) Assume suitable data, if necessary.

Q.1 **a)** *What are the merits and demerits of procedural programming language ?* **(Refer Q.2 of Chapter - 1)** **(5)**

b) *State and explain various characteristics of OOP.* **(Refer Q.7 of Chapter - 1)** **(10)**

OR

Q.2 **a)** *Explain modular programming technique.* **(Refer Q.3 of Chapter - 1)** **(3)**

b) *Explain the need of OOP.* **(Refer Q.6 of Chapter - 1)** **(4)**

c) *Explain generic programming technique and given the merits and demerits of it.* **(Refer Q.5 of Chapter - 1)** **(8)**

Q.3 **a)** *What is access modifiers ?* **(Refer Q.3 of Chapter - 2)** **(3)**

b) *Write a Java program for finding minimum of two numbers using function* **(Refer Q.4 of Chapter - 2)** **(4)**

c) *Explain call by value and call by reference.* **(Refer Q.5 of Chapter - 2)** **(8)**

OR

Q.4 **a)** *Write a short note on - this keyword.* **(Refer Q.7 of Chapter - 2)** **(5)**

b) *What is object class ?* **(Refer Q.16 of Chapter - 2)** **(3)**

c) *What is method overloading ?* **(Refer Q.8 of Chapter - 2)** **(7)**

END...

June-2022 [END SEM] [5869]-284 | **Solved Paper**

Course 2019

Time : $2\frac{1}{2}$ Hours] **[Maximum Marks : 70**

Instructions to the candidates :

1) *Answer Q.1 or Q.2, Q.3 or Q.4, Q.5 or Q.6, Q.7 or Q.8.*
2) *Neat diagrams must be drawn wherever necessary.*
3) *Figures to the right side indicate full marks.*
4) *Assume suitable data, if necessary.*

Q.1 **a)** *What are the various types of constructors in JAVA ? Explain with example.* **(Refer Q.3, Q.4 of Chapter - 3)** **(9)**

b) *What is destructor ? What is the purpose of using a destructor in C++? Explain with example.* **(9)**

Ans. :

- The destructor is called when the object is destroyed. The object can be destroyed automatically when the scope of the objects end (i.e. when the function ends) or explicitly by using operator delete.
- The destructor must have the same name as the class, but preceded with a tilde sign (~) and it must also return no value.
- We normally use the destructor when the object assigns the dynamic memory in its lifetime.

C++ Program

```
#include<iostream>
using namespace std;
class image
{
private:
        int *height,*width;
    public:
```

```
        image(int,int);//constructor
        ~image();//destructor
        int area();//regular function
};//class ends here
image::image(int x,int y)//constructor defined
{
height=new int;//assigns memory dynamically using 'new'
        width=new int;
        *height=x;
        *width=y;
}
image::~image()//destructor defined
{
delete height;//destroys the memory using 'delete'
delete width;
}
int image::area()
{
        return (*height* *width);
}
int main()
{
image obj1(10,20);
        cout<<"The area is :"<<obj1.area()<<endl;
        return 0;
}
```

Output

```
The area is : 200
```

OR

Q.2 **a)** *Design a class 'Complex' with data members for real and imaginary part. Provide default and Parameterized constructors. Write a program in JAVA to perform addition of two complex numbers.* **(9)**

Ans. : Addition of two complex numbers

```
public class Complex {
        double real, img;
        //default constructor
        Complex()
        { }
```



```
    //parameterized constructor to initialize the complex number
    Complex(double r, double i)
    {
            this.real = r;
            this.img = i;
    }
    public static Complex sum(Complex c1, Complex c2)
    {
            //creating a temporary complex number to hold
              the sum of two numbers
            Complex temp = new Complex(0, 0);
            temp.real = c1.real + c2.real;
            temp.img = c1.img + c2.img
            //returning the output complex number
            return temp;
    }
    public static void main(String args[])
    {
            Complex c1 = new Complex(5.5, 4);
            Complex c2 = new Complex(1.2, 3.5);
            Complex temp = sum(c1, c2);
      System.out.printf("Sum is: "+ temp.real+" + "+ temp.img +"i");
    }
}
```

b) *Write a short note on 'Copy Constructor' in JAVA ?* **(9)**

Ans. : Copy constructor is a special type of constructor that creates an object using another object of same Java class. It returns the duplicate copy of existing object of the class.

For example -

```
public class Student {
        private int rollno;
        private String name;
        private float marks;
        public Student(Student s){
                this.rollno = s.rollno;
                this.name = s.name;
                this.marks = s.marks;
        }
}
```

Q 3 **a)** *Define inheritance. What are the types of Inheritance? How can you inherit a class in Java ?*

(Refer Q.1, Q.4 and Q.5 of Chapter - 4) **(9)**

b) *What is polymorphism ? Illustrate types of polymorphism with example.* **(Refer Q.20 of Chapter - 4)** **(8)**

OR

Q 4 **a)** *A bank maintains two kinds of accounts for customers. One is saving and other is current. Create a class account that store customer name, account number and type of an account. Derived classes with more specific requirement and include necessary methods to achieve then following tasks :*

i) Accept deposit from the customer and update the balance.

ii) Display the balance.

iii) Compute and deposit interest.

iv) Permits withdraw and update the balance.

v) Check minimum balance condition and display necessary notice.

(Refer Q.23 of Chapter - 4) **(9)**

b) *What is interface in java ? How to declare an interface, write a syntax ? Can we achieve multiple inheritance by using interface? Justify with an example.*

(Refer Q.14, Q.15 and Q.16 Chapter - 4). **(8)**

Q.5 **a)** *What is an exception ? List out the keywords for exception handling. Write steps to develop user defined exceptions.* **(9)**

Ans. : Refer Q.1 and Q.11 of Chapter - 5.

Keywords used in Exception -

1) try 2) catch 3) throw 4) throws

b) *Write a generic method to count the number of elements in a collection that have a specific properties like odd integers, prime numbers and palindrome.* **(9)**

Ans. :

Step 1 :

```
public final class Algorithm {
  public static <T> int countIf(Collection<T> c, UnaryPredicate<T> p) {
    int count = 0;
    for (T elem : c)
      if (p.test(elem))
        ++count;
    return count;
  }
}
```

Step 2 : Define the generic **UnaryPredicate** interface.

```
public interface UnaryPredicate<T> {
  public boolean test(T obj);
}
```

Step 3 : The following program counts the number of odd integers in an integer list :

```
import java.util.*;
class OddPredicate implements UnaryPredicate<Integer> {
  public boolean test(Integer i) { return i % 2 != 0; }
}
```

Step 4 : The main program that calls the above functions is as follows -

```
public class Test {
  public static void main(String[] args) {
    Collection<Integer> ci = Arrays.asList(1, 2, 3, 4);
    int count = Algorithm.countIf(ci, new OddPredicate());
    System.out.println("Number of odd integers = " + count);
  }
}
```

OR

Q.6 **a)** *Explain linked list class with an example.*

(Refer Q.17 of Chapter - 5) **(9)**

b) *Write a program for integer division. The user enters two numbers through command line arguments as Num1 and Num2, perform division. If Num2 is Zero, an arithmetic exception must be generated.* **(9)**

Ans. :

```
class Test
{
	public static void main(String[] args)
	{
		int a = Integer.parseInt(args[0]);
		int b = Integer.parseInt(args[1]);
		int ans = 0;
		try {
			ans= a/b;
			System.out.println("The result is: "+ans);
		}
		catch(ArithmeticException e)
		{
			System.out.println("Divide by zero");
		}
	}
}
```

Q.7 **a)** *What is stream ? Explain various stream classes in Java.*

(Refer Q.1, Q.2 and Q.3 of Chapter - 6) **(9)**

b) *Write a Java program to copy the content of the file "file1.txt" into new file "file2.txt"*

(Refer Q.8 of Chapter - 6) **(8)**

OR

Q.8 **a)** *Write a short note on :* **(8)**

i) Adaptor **(Refer Q.15 of Chapter - 6)**

ii) Singleton **(Refer Q.16 of Chapter - 6)**

b) *Implement a program for maintaining a database of student records using* ***Files****. Student has Student_id, name, Roll_no, Class, marks and address.*

Display the data for few students. **(9)**

i) Create Database. ii) Display Database.

iii) Delete Records. iv) Update Record.

v) Search Record. **(Refer Q.18 of Chapter - 6)**

END ...