THOMAS BASYAL
BEIT (DAY)
ROLLNO=181546

SECTION 3.
SOLUTION:

Shape.java

```
1   Package com.math;
2
3   class Shape{
4       private int xaxis;
5       private int yaxis;
6       private int length;
7       private int width;
8
9       public Shape(int x,int y, int l, int w){
10          this.xaxis=x;
11          this.yaxis=y;
12          this.length=l;
13          this.width=w;
14      }
15      public Shape(int l, int w){
16          this(0,0,l,w);
17      }
18      public Shape(){
19          this(0,0,1,1);
20      }
21  }
22
23  public class Main{
24      public static void main(String[] args){
25          Shape o1=new Shape(1,7,6,8);
26          Shape o2=new Shpae(5,2);
27          Shape o3=new shape();
28      }
29  }
```

SECTION 2

QUESTION NO 1.2.3.

Shape.java | Student.java

```
1   Package Thomas181546;
2
3   class Student{
4       private String name;
5       private int roll;
6       private String college;
7
8       public void putData(String n, int r, String c){
9           this.name=n;
10          this.roll=r;
11          this.college=c;
12      }
13      public void showData(){
14          System.out.println("Name:"+name+"\n"+"Roll no:"+roll+"\n"+"College:"+college);
15      }
16  }
17  public class Main{
18      public static void main(String[] args){
19          Student o=new Student();
20          o.putData("Thomas", 181546, "NCIT");
21          o.showData();
22      }
23  }
```

QUESTION NO 4 USING CONSTRUCTOR

Shape.java | Student.java

```
1  Package Thomas181546;
2
3  class Student{
4      private String name;
5      private int roll;
6      private String college;
7
8      public Student(String n, int r, String c){
9          this.name=n;
10         this.roll=r;
11         this.college=c;
12     }
13     public void showData(){
14         System.out.println("Name:"+name+"\n"+"Roll no:"+roll+"\n"+"College:"+college);
15     }
16 }
17 public class Main{
18     public static void main(String[] args){
19         Student o=new Student("Thomas", 181546, "NCIT");
20         o.showData();
21     }
22 }
```

QUESTION NO 5

Shape.java | Student.java

```
1   Package Thomas181546;
2
3   class Student{
4       private String name;
5       private int roll;
6       private String college;
7       static private int counter=0;
8
9
10      public Student(String n, int r, String c){
11          this.name=n;
12          this.roll=r;
13          this.college=c;
14          counter+=1;
15      }
16      public void showData(){
17          System.out.println("Name:"+name+"\n"+"Roll no:"+roll+"\n"+"College:"+college);
18      }
19      static void total(){
20          System.out.println("Number of a object that exists is"+counter);
21      }
22  }
23  public class Main{
24      public static void main(String[] args){
25          Student o1=new Student("Thomas", 181546, "NCIT");
26          Student o2=new Student("suraj", 181544, "NCIT");
27          Student o3=new Student("ujhwal", 181547, "NCIT");
28          o1.showData();
29          o2.showData();
30          o3.showData();
31          Student.total();
32      }
33  }
```

JAVA ASSIGNMENT 1 COMPLETED

3) Write a program to handle the following Exceptions.

i) ArithmeticException

```
{
   public static void main(String args[])
   {
      try{
         int num1=30, num2=0;
         int output=num1/num2;
         System.out.println ("Result: "+output);
      }
      catch(ArithmeticException e){
         System.out.println ("You Shouldn't divide a number by zero
```

ii) ArrayIndexOutOfBoundsException

```
public class ArExp {

   public static void main(String[] args) {
      try{
String[] arr = {"thomas","bhusansir","asbin","aayush"};


      for(int i=0;i<=arr.length;i++) {


            System.out.println(arr[i]);

      } catch(ArrayIndexOutOfBoundException e){
       e.PrintStackTrace();
      }
   }
}
```

iii) NullPointerException

```
class Exception2
{
   public static void main(String args[])
   {
        try{
                String str=null;
                System.out.println (str.length());
        }
        catch(NullPointerException e){
                System.out.println("NullPointerException..");
        }
   }
}
```

iv) NumberFormatException

```
class ArExp
{
   public static void main(String args[])
   {
      try{
    int num=Integer.parseInt ("XYZ") ;
    System.out.println(num);
      }catch(NumberFormatException e){
    System.out.println("Number format exception occurred");
      }
  }
```

4. Write a program to showcase how nested try and catch can be used. Also use finally

```
// main method
public static void main(String args[])
{
    // Main try block
    try {
```

```
        // initializing array
        int a[] = { 1, 2, 3, 4, 5 };

        // trying to print element at index 5
        System.out.println(a[5]);

        // try-block2 inside another try block
        try {

            // performing division by zero
            int x = a[2] / 0;
        }
        catch (ArithmeticException e2) {
            System.out.println("division by zero is not possible");
        }
    }
    catch (ArrayIndexOutOfBoundsException e1) {
        System.out.println("ArrayIndexOutOfBoundsException");
        System.out.println("Element at such index does not exists");
    }finally{
        System.out.println("Description of all exceptions here");
    }
}
}
```

6) Create a class Student that has Name(String), semester(int) and grade(float). Create a custom exception class named SemesterException and GradeException that will be thrown if the user try to create a student object with invalid semester or grade.

```
package Bhusansir;
public class Student{
    String name;
    int semester;
    float grade;
    public Student(String name, int semester, float grade){
        this.name=name;
        this.semester=semester;
```

```
        this.grade=grade;
    if(semester<1 || semester>8){
    throw new SemesterException("semesester is less then 1 or greater than 8");
    }
     if(grade>4.0){
       throw new GradeException("grade is more than 4.0");
    }
}
   public void getData(){
       System.out.println(this.name);
       System.out.println(this.semester);
       System.out.println(this.grade);

   }
   public static void main(String[] args){
       try{
           Student student=new Student("thomas", 4, 7.6);
           student.getData();

       }catch(SemesterException e){
           System.out.println(e.getMessage());
       }catch(GradeException m)
       {
           System.out.println(m.getMessage());
       }
   }
}
```

```
package Bhusansir;
public class GradeException extends Exception{
   public GradeException(String message){
       super(message);
   }
}
```

```
package Bhusansir;
class SemesterException extends Exception{
    public SemesterException(String s){
        super(s);
    }

}
```

2. Write a program to copy the content of one txt file to another using both Character stream class and Byte stream class.

I. Using Character Stream Class

```
import java.io.FileReader.*;

public class CopyCharacters {
    public static void main(String[] args) throws IOException {

        FileReader inputStream = null;
        FileWriter outputStream = null;

        try {
            inputStream = new FileReader("kathmandu.txt");
            outputStream = new FileWriter("pokhara.txt");

            int c;
            while ((c = inputStream.read()) != -1) {
                outputStream.write(c);
            }
        } finally {
            if (inputStream != null) {
                inputStream.close();
            }
            if (outputStream != null) {
                outputStream.close();
            }
        }
    }
}
```

II. Using Byte Stream Class

```
[redacted]Bytess{
[redacted]([redacted][][redacted])[redacted]{
[redacted]=[redacted]("kathmandu.txt");
[redacted]=[redacted]("pokhra.txt");
[redacted];
while((by=fis.read())!=[redacted]){
[redacted]([redacted]);
}
[redacted]();
[redacted]();
[redacted]}
}
```

4. Write a program to take name(String), grade(float) and faculty(String) from the user and save them in a file "student.dat". User should be prompted after each iteration. Also catch all suitable exceptions.

```java
import java.io.DataOutputStream;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.util.Scanner;

public class Student {

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        try {

            FileOutputStream f = new FileOutputStream("student.dat");
            DataOutputStream d = new DataOutputStream(f); // can write any data types using dataoutputstream

            while (true) { // for infinite loop

                System.out.println("Enter name");
                String name = sc.next();

                System.out.println("Enter grade");
                float grade = sc.nextFloat();

                System.out.println("Enter faculty");
                String faculty = sc.next();

                d.writeUTF(name);
                d.writeFloat(grade);
                d.writeUTF(faculty);
                System.out.println("Press any key to continue, press N to exit");

                String prompt = sc.next();
                if (prompt.equalsIgnoreCase("n")) {
                    break; // loop stops if user enters N
                }

            }
            f.close();
            d.close();


        } catch (FileNotFoundException e) {

            System.out.println("file not found!!");
        } catch (Exception e) {
            System.out.println(e.getMessage());

        }
    }

}
```

5. Write a program that reads the "student.dat" file created in previous question and display the record of only those students have passed.

```
import java.io.DataInputStream;
import java.io.FileInputStream;
import java.io.FileNotFoundException;

public class StudentRead {

    public static void main(String[] args) {
        try {
            FileInputStream f = new FileInputStream("student.dat");
            DataInputStream d = new DataInputStream(f);

            while (d.available() > 0) {


                String name = d.readUTF();
                float grade = d.readFloat();
                String faculty = d.readUTF();


                if (grade > 2.0) {
                    System.out.println("Name " + name);
                    System.out.println(" grade " + grade);
                    System.out.println(" faculty " + faculty);

                }

            }
            f.close();
```

6. Write a program that takes user input from the console and displays it back to the console using following three different methods:

Using console

```
public class Consoles{
    public static void main(String[] args) {
        System.out.print("Enter your Job: ");
        String job = System.console().readLine();
        System.out.println("You are a " + job);
    }
}
```

Using Command-Line arguments

Using inputstream

```
import java.io.*;

public class Test {
    public static void main(String[] args) throws IOException {


        BufferedReader rdr = new BufferedReader(new
InputStreamReader(System.in));

        // Display message for user
        System.out.print("Enter your Job: ");

        String job = rdr.readLine();
        System.out.println("You are a " + job);

    }
}
```

Using scanner

8. Create a simple class Student with at least two instance variables and a method. Write a program to create the object of the class Student and save it in a "student.txt" file.

```
package FileReadWrite;

import java.io.Serializable;

public class Student implements Serializable {

    int roll;
    String name;

    Student(int roll, String name) {

        this.roll = roll;
        this.name = name;

    }

    void show() {
        System.out.println("Name: " + this.name + " Roll: " + this.roll);
    }

}
```

```
package FileReadWrite;

import java.io.FileOutputStream;
import java.io.ObjectOutputStream;

public class WriteObjectStudent {

    public static void main(String[] args) {

        Student t = new Student(1, "Manita");
        try {

            FileOutputStream f =new FileOutputStream("student.txt");
            ObjectOutputStream obj = new ObjectOutputStream(f);
            obj.writeObject(t);
            f.close();
            obj.close();

         }catch(Exception e) {

         }


    }
```

9. WAP to the read the object stored in the "student.txt" file in the previous question and display the state of the object

```
import java.io.*;

classs StuObj{

public static void main(String[] args){


try{
   FileInputStream fin=new FileInputStream("Student.txt");
   ObjectInputStream o=new ObjectInputStream(fin);
Student s=(Student)o.readObject();
   s.show();
   fin.close();
```

```
        o.close();
      }
      catch(Exception e){
      e.PrintStackTrace();
     }
    }
   }
```

10. WAP to archive a file into a zip file and also read the content of the zip file.

```
package FileReadWrite;

import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.util.zip.ZipEntry;
import java.util.zip.ZipOutputStream;

public class ZipTest {

    public static void main(String[] args) {
        try {

            FileInputStream in = new FileInputStream("student.txt");
            FileOutputStream out = new FileOutputStream("student.zip");

            ZipOutputStream zout = new ZipOutputStream(out);
            ZipEntry ze = new ZipEntry("student.zip");
            zout.putNextEntry(ze);

            int content;
            while ((content = in.read()) != -1) {

                zout.write((byte) content);

            }
            zout.close();
            in.close();
            out.close();

        } catch (Exception e) {

        }

    }

}
```

11. WAP to archive a folder into a compressed folder (zip folder).

```
import java.io.*;

import java.util.zip.*;
;

public class Zipper {

public static void main(String[] args) {
String result_message=zipFiles("D:/zipped_folder.zip","D:/files/");
System.out.println(result_message);

}

public static String zipFiles(String zipFile, String srcDir) {
try {
File srcFile = new File(srcDir);
File[] files = srcFile.listFiles();
FileOutputStream fos = new FileOutputStream(zipFile);

ZipOutputStream zos = new ZipOutputStream(fos);

for (int i = 0; i < files.length; i++) {

byte[] buffer = new byte[1024];

FileInputStream fis = new FileInputStream(files[i]);

zos.putNextEntry(new ZipEntry(files[i].getName()));

int length;

while ((length = fis.read(buffer)) > 0) {
zos.write(buffer, 0, length);
}

zos.closeEntry();

fis.close();

}
zos.close();
}
//
catch (Exception e)
{return e.getMessage();}
return "Successfully created the zip file"+zipFile;
}
```

NAME: THOMAS BASYAL FACULTY: BEIT DAY ROLLNO: 181546

2014 spring 2a) solution by THOMAS BASYAL

Implement an abstract class named Person and two subclasses named Student and Employee in Java. A person has a name, address, phone number and e-mail address. A student has a class status (freshman, sophomore, junior, or senior). Define the status as constant. Employee has an office, salary, and date-hired. Implement the above classes in Java. Provide Constructors for classes to initialize private variables. Override the toString() method in each class to display the class name and the person's name. Write an application to create objects of type Student and Employee, and print the person's name and the class name of the objects.

ANS:

```
abstract class Person
{
      protected String name;
      protected String address;
      protected String emailID;
      protected String phoneNo;

public Person(String name, String address, String phoneNo, String emailID)
{
      this.name = name;
      this.address = address;
      this.phoneNo=phoneNo;
      this.emailID=emailID;
 }

}
class Student extends Person
{
     private final String STATUS = "freshman";
     Student(String name,String address,String phoneNo,String emailID)
     {
       super(name,address,phoneNo,emailID);
     }
public String toString()
 {
      return this.name+" "+getClass().getName();
 }
}
class Employee extends Person
{
private String office;
```

```
private double salary;
private String dateHired;
Employee(String name, String address, String phoneNo, String emailID, String off
ice, double salary, String dateHired)
 {
     super(name, address, phoneNo, emailID);
     this.office=office;
     this.salary=salary;
     this.dateHired=dateHired;
 }.
 public String toString()
  {
      return this.name+" "+getClass().getName();
  }
}
class Thomasbasyal
  {
 public static void main(String[] args)
  {
   Employee employee = new Employee("bhusan sir","america","9818911707","bhusan@n
cit.edu.np","apple company",205200000,"22 may");
   Student student = new Student("thomas basyal","canada","9863444041","basyaltho
mas81@gmail.com");
   System.out.println(student);
   System.out.println(employee);

 }
}
```