Assignment 2

Bollepalli Sai Sreekanth

1.Write a Java program to show the use of all keywords for exception

Handling.

Code:

```
package sre;

import java.util.Scanner;

public class Main11 {

	public static void main(String[] args) {

		Scanner sc=new Scanner(System.in);
		int a=sc.nextInt();
		try {
			if(a==1) {
				a=10/0;
			}
			else {
				throw new ();
			}

		}
		catch(ArithmeticException e) {
			System.out.println("Arithmatic Exception");
		}

		catch(NullPointerException e) {
			System.out.println("NullPointerException");

		}
		finally {

	    System.out.println("finally : it will execute.");

	  }
	}

}
```

Output:

```
1
Arithmatic Exception
finally : it will execute.
2
NullPointerException
finally : it will execute.
```

2. Write a Java program using try and catch to generate NegativeArrayIndex Exception and Arithmetic Exception.

A.

Code:

```
package sre;

import java.util.Scanner;

public class Main21 {

	public static void main(String[] args) {
		Scanner sc=new Scanner(System.in);
	try {
		System.out.println("Enter the Size of array");
		int size=sc.nextInt();
		if(size<=0) {
		int[] arr=new int[size];
		}
		System.out.println("Enter a number to divide with size");
		int b=sc.nextInt();
		 b=size/b;

	}
	catch(NegativeArraySizeException e){
		System.out.println("Array size can't be Negative" +e);

	}
	catch(ArithmeticException e) {
		System.out.println("any number cannot be divided with zero "+e);
	}


	}

}
```

Output:

1.

```
Enter the Size of array
-5
Array size can't be Negative java.lang.NegativeArraySizeException: -5
```

2.

```
Enter the Size of array
5
Enter a number to divide with size
0
any number cannot be divided with zero java.lang.ArithmeticException: / by zero
```

3. Write a class that keeps a running total of all characters passed to it (one at a time) and throws an exception if it is passed a non-alphabetic character.

A.

Code:

```
package sre;

import java.util.Scanner;
class NotAAlphabeticCharException extends Exception{

	public NotAAlphabeticCharException(String string) {

	}

}

public class Main23 {

	public static void main(String[] args) {
		try {
		Scanner sc=new Scanner(System.in);
		System.out.println("Enter a word");
		String a=sc.nextLine();
		for(int i=0;i<a.length();i++) {
			if(Character.isLetter(a.charAt(i))) {
				continue;
			}

			else {
				throw new
NotAAlphabeticCharException("NotAAlphabeticCharException");
			}
		}}
		catch(NotAAlphabeticCharException e){
			System.out.println("only letters should be entered "+ e);

		}

	}

}
```

Output:

```
Enter a word
sreekanth1
only letters should be entered sre.NotAAlphabeticCharException
```

4. Write a class that keeps a running total of all characters passed to it (one at a time) and throws an exception if it is passed a non-alphabetic character is passed.

A.

Code:

```
package sre;

import java.util.Scanner;
class NotAAlphabeticCharException extends Exception{

	public NotAAlphabeticCharException(String string) {

	}

}

public class Main23 {

	public static void main(String[] args) {
		try {
		Scanner sc=new Scanner(System.in);
		System.out.println("Enter a word");
		String a=sc.nextLine();
		for(int i=0;i<a.length();i++) {
			if(Character.isLetter(a.charAt(i))) {
				continue;
			}

			else {
				throw new
NotAAlphabeticCharException("NotAAlphabeticCharException");
			}
		}}
		catch(NotAAlphabeticCharException e){
			System.out.println("only letters should be entered "+ e);

		}

	}

}
```

Output:

Enter a word
sreekanth1
only letters should be entered sre.NotAAlphabeticCharException

5. Create a user-defined exception named CheckArgument to check the number of arguments passed through the command line. If the number of arguments is Page: 9 less than 5, throw CheckArgumentexception, else print the addition of all the five numbers.
A.

Code:

```
package sre;

class CheckArgumentexception extends Exception{
   CheckArgumentexception(String s) {
      super(s);
   }
}
public class Main25 {
   public static int calculateSum(int args,int[] arr) throws
CheckArgumentexception{
      int sum=0;
      if(args<5 )
         throw new CheckArgumentexception("The Number of Arguments
passed in the CLI is less than 5");
      else if(args>5) {
       throw new CheckArgumentexception("The Number of Arguments
passed in the CLI is more than 5");
      }
      else{
         for(int i=0;i<args;i++){
            sum=sum+arr[i];
         }
      }
      return sum;
   }
   public static void main(String[] args) {

      int[] arr = new int[args.length];
      for(int i=0;i<args.length;i++){
       arr[i]= Integer.parseInt(args[i]);
       }

      try{
         int result = calculateSum(args.length,arr);
         System.out.println("The Sum of the numbers is: "+result);
      }
```

```
        catch(CheckArgumentexception e){
            System.out.println(e);
        }

    }
}
```

Output:

1.input java Main25 1 2 3 4 5

```
The Sum of the numbers is: 15
```

2.input java Main25

```
sre.CheckArgumentexception: The Number of Arguments passed in the CLI is less than
5
```

6. Design an abstract class having two methods. Create Rectangle and Triangle classes by inheriting the shape class and override the above methods to suitably implement for Rectangle and Triangle class.

A.

Code:

```
package sre;
abstract class shape{
        double height;
        double width;
        public shape(int i, int j) {
                this.height=i;
                this.width=j;
        }
        abstract double area();
        abstract double show();

}
class Rectangle extends shape{



        public Rectangle(int i, int j) {
                super(i,j);
        }

        @Override
        double area() {
                System.out.println("Area : "+width*height);
                return 0;
        }
```

```
	@Override
	double show() {
		System.out.println("Rectangle");
		return 0;
	}

}
class Traingle extends shape{

	public Traingle(int i, int j) {
		super(i, j);

	}

	@Override
	double area() {
		System.out.println("Area : "+0.5*width*height);
		return 0;
	}

	@Override
	double show() {
		System.out.println("Triangle");
		return 0;
	}

}
public class Main26 {

	public static void main(String[] args) {
		Rectangle r=new Rectangle(8,9);
		r.show();
		r.area();
		Traingle r1=new Traingle(8,9);
		r1.show();
		r1.area();

	}

}
```

Output:

```
Rectangle
Area : 72.0
Triangle
Area : 36.0
```

7. Write a Java program to explain “Enum.”

A.

Code:

```
package sre;
enum Monuments{
	Charminar,
	Tajmahal,
	GoldenTemple,
	AjantaCaves

}
public class Main27 {

	public static void main(String[] args) {
		Monuments m=Monuments.Charminar;
		switch(m){
			case Charminar:
				System.out.println("Charminar is located in
Hyderabad");
				break;
			case Tajmahal:
				System.out.println("Tajmahal is located in Delhi");
				break;
			case AjantaCaves:
				System.out.println("AjantaCaves is located in
Maharashtra");
				break;
			case GoldenTemple:
				System.out.println("GoldenTemple is located in
Amritsar");
				break;

		}

	}

}
```

Output:

```
Charminar is located in Hyderabad
```

8. Write a Java program for user role management with Enum.

A.

Code:

```
package sre;
enum Monuments{
	User,
	Admin,
	Developer

}
public class Main27 {

	public static void main(String[] args) {
		Monuments m=Monuments.Admin;
		switch(m){
			case Developer:
				System.out.println("Welcome Developer");
				break;
			case Admin:
				System.out.println("Welcome Admin");
				break;
			case User:
				System.out.println("Welcome User");
				break;


		}

	}

}
```

Output:

```
Welcome Admin
```

9. Write a Java program for user role management with Autoboxing & Unboxing

A.

Code:

```
package sre;

import java.util.ArrayList;

public class Main29 {

	public static void main(String[] args) {
		ArrayList<Object> userManagement=new ArrayList<Object>();
		userManagement.add("Admin");
		userManagement.add("User");
		userManagement.add("Developer");
		System.out.println(userManagement);

	}
```

```
}
```
Output:
```
[Admin, User, Developer]
```

10. Write a Java Program to explain the Generic Methods.

A.

Code:
```
package sre;
class Tes<T, U>
{
	T obj1;
	U obj2;
	Tes(T obj1, U obj2)
	{
		this.obj1 = obj1;
		this.obj2 = obj2;
	}
	public void show()
	{
		System.out.println(obj1);
		System.out.println(obj2);
	}
}
class Main1{
	public static void main (String[] args)
	{
		Tes <String, Integer> obj =
			new Tes<String, Integer>("Sreekanth",7);
		obj.show();
	}
}
```

Output:
```
Sreekanth
7
```