MODULE - 3

1. Write a program to read file, encrypt it (encryption key is +3), and store encrypted data to another file.(Use FileInputStream/OutputStream).

CODE:

```
import java.io.FileInputStream;
import java.io.FileOutputStream;

public class Encrypt{
        static int encrypt(int data , int key){
                data = data +key;
                return data;
}

 public static void main(String[] args) {

 FileInputStream fin = null;
 FileOutputStream fout = null;

 try {
 fin = new FileInputStream("sample.txt");
 fout = new FileOutputStream("enc.txt");
 int data =0;

 while(fin.available() >0){
 data = fin.read();
 fout.write((char)(encrypt(data, 3)));
 }
 } catch (Exception e) {
 System.out.println(e);
 }
System.out.println("DATA IS ENCRYPTED");
 }
}
```

OUTPUT :

```
javac Encrypt.java
java Encrypt
DATA IS ENCRYPTED
```

Content in “sample.txt” : - JAVA is great
Content in “enc.txt” after encryption :- MDYD#lv#juhdw

2. Write a program, which reads encrypted file generated in 1st program, decrypt it, print it to the console.(Use FileInputStream).

CODE:

```
import java.io.FileInputStream;

public class Decrypt{

 static int decrypt(int data , int key){
 return data = data - key;
 }
 public static void main(String[] args) {
 FileInputStream fin = null;
 try {
 fin = new FileInputStream("enc.txt");

 int data =0;

 System.out.println("Decrypted Data is :");
 while(fin.available() >0){
 data = fin.read();
 System.out.print((char)(decrypt(data, 3)));
 }


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

OUTPUT :

```
javac Decrypt.java
java Decrypt

Decrypted Data is :
JAVA is great
```

3. Write a program to Copy the data of a file, file name provided from command line argument, to another file.(Use Buffered I/O Stream).

CODE:

```
import java.io.BufferedInputStream;

import java.io.BufferedOutputStream;

import java.io.FileInputStream;

import java.io.FileOutputStream;
```

```
import java.io.IOException;



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

      BufferedInputStream bis = null;
      BufferedOutputStream bos = null;
      FileInputStream fis = null;
      FileOutputStream fos = null;

      try {
               fis = new FileInputStream("chill.txt");
               bis = new BufferedInputStream( fis);
               fos = new FileOutputStream(args[0]);
               bos = new BufferedOutputStream(fos);

               int b;
               while ((b = bis.read()) != -1) {
                        bos.write(b);
               }
               bos.flush();
      }
      catch(IOException ex) {
               System.err.println(ex.getMessage());
      }

      finally {
               if(fis!=null) fis.close();
               if(bis!=null) bis.close();
               if(fos!=null) fos.close();
```

```
            if(bos!=null) bos.close();
        }


        System.out.println("File has been Copied");
}
}
```

OUTPUT:

javac CopyaTob.java

java CopyaTob write.txt

File has been Copied

4. Write a ProductManagement program, which will store 5 objects of Product class to a file name “ProductDetails.dat”.(Use ObjectOutputStream)

CODE:

```
import java.io.*;

class Product implements Serializable{

        private int pId;
        private String prName;
        private int pRate;

        static int count;

        public Product(String pName, int rate){
                pId = 101 + count++;
                this.prName = pName;
                pRate = rate;
        }

        public Product(String name){
                pId = 101 + count++;
                this.prName = name;
                pRate = 100;
        }

        public Product(){
                this("NoName", 0);
        }

        public Product(Product p){
```

```
		prName = p.prName;
		pRate = p.pRate;
	}

	public String getName(){
		return prName;
	}

	public void setId(int id){
		pId = id;
	}

	public void setRate(int rate){pRate = rate;}

	public void setName(String name) {prName = name;}

	public int getId(){return pId;}

	public int getRate(){return pRate;}

	public String toString(){
		return " Product id : " + pId + ", name : " + prName + " and rate : " + pRate;
	}

	public boolean equals(Object other){
		System.out.println("Equals method!");
		if(other instanceof Product){
			Product p = (Product) other;
			return (this.prName == p.prName) && (this.pRate == p.pRate);
		}
		return false;
	}
}

class ObjectInputOutputStream{

	public static void main(String[] args) throws Exception{

		Product p1 = new Product("p1", 200);
		Product p2 = new Product("p2", 100);
		Product p3 = new Product("p3", 50);

		ObjectOutputStream os = new ObjectOutputStream(new
FileOutputStream("ProductDetails.txt"));

		ObjectInputStream or = new ObjectInputStream(new
FileInputStream("ProductDetails.dat"));
```

```
                os.writeObject(p1);
                os.writeObject(p2);
                os.writeObject(p3);

                Product p =(Product) or.readObject();
                try{
                        while(p != null){
                                System.out.println(p);
                                p = (Product) or.readObject();
                        }
                }catch(Exception e){}

                os.close();
                or.close();
        }
}
```

OUTPUT :

```
javac ObjectInputOutputStream.java
java ObjectInputOutputStream

Product id : 101, name : p1 and rate : 200
 Product id : 102, name : p2 and rate : 100
 Product id : 103, name : p3 and rate : 50
```

Content in "ProductDetails.dat" created after running code:-
¬í sr Productø³ ï"±ey I pIdI pRateL prNamet Ljava/lang/String;xp e Èt p1sq ~ f dt p2sq ~ g 2t p3

5. Write a program, Which will read file "ProductDetails.dat" (created by above program).Print the details of the object having highest price and lowest price. (Use ObjectInputStream)

CODE:

```
import java.io.FileInputStream;
import java.io.ObjectInputStream;
import java.io.Serializable;

class Product implements Serializable{
 protected int pId;
 protected String pName;
 protected int pRate;
 static int count;
 {
 count++;
 }
 public static void getProductCount(){ System.out.println(count);}
 public String getName(){return pName; }
 public void setpName(String pName){ this.pName= pName;}
```

```
 public int getpId(){ return pId;}
 public void setpId( int id){ pId = id;}
 public int getpRate(){return pRate;}
 public int getCount(){ return count;}
 public Product(String name , int rate , int id){
 pName = name;
 pRate = rate;
 pId = id;
 }
 public String toString(){
 return " Product id is " + pId + " name is " +pName ;
 }

 public Product() {
 }
}
public class ProducthTol {
 public static void main(String[] args) {
 try {
 FileInputStream fileIn = new FileInputStream("ProductDetails.dat");
 ObjectInputStream objIn = new ObjectInputStream(fileIn);

 Product newPr = (Product) objIn.readObject();
 Product newPr1 = (Product) objIn.readObject();
 Product newPr2 = (Product) objIn.readObject();
 System.out.println(" Product Name: " + newPr.getName());
 System.out.println(" Rate: " + newPr.getpRate());
 System.out.println(" Product Name: " + newPr1.getName());
 System.out.println(" Rate: " + newPr1.getpRate());
 System.out.println(" Product Name: " + newPr2.getName());
 System.out.println(" Rate: " + newPr2.getpRate());
 objIn.close();
 }
 catch (Exception e) {
 e.getStackTrace();
 }
 }
}
```

OUTPUT :

```
Product id : 101, name : p1 and rate : 200
Product id : 103, name : p3 and rate : 50
```

6. Write a program which will read a text file and print the count of total number of Lines, Words and Characters in it. (Use BufferedReader).

CODE:

```
import java.io.BufferedReader;
import java.io.FileReader;
import java.io.IOException;

public class CountFile
{
   public static void main(String[] args)
   {
      BufferedReader reader = null;

      int lineCount = 0;
      int wordCount = 0;
      int charCount = 0;

      try
      {
         reader = new BufferedReader(new FileReader("sample.txt"));
         String currentLine = reader.readLine();

         while (currentLine != null)
         {
            lineCount++;

            String[] words = currentLine.split(" ");

            wordCount = wordCount + words.length;

            for (String word : words)
            {
               charCount = charCount + word.length();
            }

            currentLine = reader.readLine();
         }

         System.out.println("Number Of Lines In A File : "+lineCount);
         System.out.println("Number Of Words In A File : "+wordCount);
         System.out.println("Number Of Chars In A File : "+charCount);

      }
      catch (IOException e)
      {
         e.printStackTrace();
      }
      finally
      {
         try
         {
```

```
            reader.close();
          }
          catch (IOException e)
          {
            e.printStackTrace();
          }
        }
      }
    }
```

OUTPUT :

```
javac CountFile.java
java CountFile

Number Of Lines In A File : 3
Number Of Words In A File : 8
Number Of Chars In A File : 30

Contents in file “sample.txt” :-
Hi
I am learning java
java is awesome
```

7. Write a program to read a text file and copy it’s content in uppercase form to another file.(Use BufferedWriter\Reader)

CODE:

```
import java.io.*;
import java.util.*;



class CopyUppercase {
    public static void main(String[] args) {
        String inputFilePath = "./input.txt";
        String outputFilePath = "./output.txt";
        capitalizeFile(inputFilePath, outputFilePath);
    }

    static void capitalizeFile(String inputFilePath, String outputFilePath) {
```

```
        try {
                BufferedReader br = new BufferedReader(new FileReader(inputFilePath));
                PrintWriter pr = new PrintWriter(new FileWriter(outputFilePath));
                String s;
                for(s = br.readLine(); s != null; s = br.readLine())
                        pr.println(s.toUpperCase());
                System.out.println("File written successfully");
                br.close();
                pr.flush();
                pr.close();

        } catch(Exception e) { System.out.println("File error: " + e); }
    }
}
```

OUTPUT :

```
javac CopyUppercase.java
java CopyUppercase
File written successfully
```

Contents in the file "input.txt" :- hi lets go to south korea it would be fun

Contents in the file "output.txt" :- HI LETS GO TO SOUTH KOREA IT WOULD BE FUN

8. Write a program to copy content of a file while removing duplicates lines

CODE:

```
import java.io.*;
import java.util.*;

class DuplicateDelete {
        public static void main(String[] args) {
                String filePath = "./javaDUP.txt";
```

```
            deleteDuplicates(filePath);
        }

        static void deleteDuplicates(String filePath) {
            try {
                BufferedReader br = new BufferedReader(new FileReader(filePath));
                Set<String> uniqueLines = new LinkedHashSet<String>();
                String s;
                for(s = br.readLine(); s != null; s = br.readLine())
                    uniqueLines.add(s);
                br.close();
                PrintWriter pw = new PrintWriter(new FileWriter(filePath));
                for(String s_: uniqueLines)
                    pw.println(s_);
                pw.flush();
                pw.close();
                System.out.println("Duplicate lines removed successfully");
            } catch(Exception e) {
                System.out.println("File error: " + e);
            }
        }
}
```

OUTPUT :

```
javac DuplicateDelete.java
java DuplicateDelete
Duplicate lines removed successfully

Content in file before operation :- hi i am java
                                   hi i am java
                                   java is good

Content in file before operation :- hi i am java
                                   java is good
```

9. Write an Exception handling program, which will handle RuntimeException, ArrayIndexOutOfBoundsException, NumberFormatException, ArithmeticException, NullPointerException.(Use Multiple catch with single try block).

CODE:

```
class MultipleCatch{
```

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

		System.out.println("Start of main");
		try{
			int number1 = Integer.parseInt(args[0]);
			int number2 = Integer.parseInt(args[1]);
			String name = null;
			if(name == null)
				throw new NullPointerException();
			int number3 = number1 / number2;
			System.out.println(number1 + "\t" + number2 + "\t" + number3);

		}
		catch(ArrayIndexOutOfBoundsException e){
			e.printStackTrace();
			System.out.println("Please provide command line argument!!");
		}catch(ArithmeticException e){
			 System.out.println("A number cannot be divided by zero");
		}catch(NumberFormatException e){
			System.out.println("Only integer argumnet required!!!");
		}catch(RuntimeException e){
			System.out.println(e);
		}catch(Exception e){
			System.out.println("Null Reference");
			e.printStackTrace();
			System.out.println(e.getMessage());
			System.out.println(e);
		}
		finally{
			System.out.println("finally block");
		}

		System.out.println("End of main");
	}
}
```

OUTPUT :

```
javac MultipleCatch.java
 java MultipleCatch 2 4

Start of main
java.lang.NullPointerException
finally block
End of main
```

10. Write a program, to demonstrate nested try-catch-finally structure.

CODE:

```
class NestedTryCatch
{
   public static void main(String arg[])
   {
      int arr[]=new int[5];
      try
      {

         try
         {
            System.out.println("Divide 1");
            int b=23/0;
         }
         catch(ArithmeticException e)
         {
            System.out.println(e);
         }
         try
         {
            arr[7]=10;
            int c=22/0;
            System.out.println("Divide 2 : "+c);
         }
         catch(ArithmeticException e)
         {
            System.out.println("Err:Divide by 0");
         }
         catch(ArrayIndexOutOfBoundsException e)

         {
            System.out.println("Err:Array out of bound");
         }
      }
      catch(Exception e)
      {
         System.out.println("Handled");
      }
   }
}
```

OUTPUT :

```
javac NestedTryCatch.java
java NestedTryCatch
Divide 1
java.lang.ArithmeticException: / by zero
Err:Array out of bound
```

11. Write a program, to create and handle user defined Unchecked Exception – InvalidBoxException which will be thrown from the constructor of the Box class, when either length or width or height of Box is less than zero.

CODE:

```
class InvalidBOXException extends Exception
{

        InvalidBOXException(String msg)
        {
                super(msg);
                System.out.println("Invalid BOX due to either length or width or height less
than zero");
        }
}
class BOX
{
        private double length;
        private double width;
        private double height;

        BOX(double l, double w, double h) throws InvalidBOXException
        {
                if( l <=0 || w <= 0 || h <= 0)
                        throw new InvalidBOXException("Invalid Box");
                length = l;
                width = w;
                height = h;
        }
        double getLength() { return length; }
        double getWidth()  { return width;  }
        double getHeight() { return height; }

        double area()   { return 2*(length*width + width*height + height*length); }
        double volume() { return length*width*height ; }
}
class ExceptionTest
{
        public static void  main(String args[])
        {
                try{
                        BOX b2 = new BOX(10,4,5);
                        System.out.println("Area of b2(10, 4, 5):"+b2.area());
```

```
                    BOX b1 = new BOX(0,0,0);


                        }catch(InvalidBOXException ib){System.out.println("Attempt to create
Invalid Box ! failed");}
            }
}
```

OUTPUT :

```
javac ExceptionTest.java
java ExceptionTest

Area of b2(10, 4, 5):220.0
Invalid BOX due to either length or width or height less than zero
Attempt to create Invalid Box ! failed
```

12. Write a program to create and handle User Defined CheckedException – InsufficientFundsException, generated while withdrawing amount more than available balance.Create necessary class and methods to support this scenario.

CODE:

```
import java.util.*;
class Bank
{
   float fund;
   void deposit(float amount)
   {
      fund=amount;
   }
   void withdraw(float money) throws Exception
   {
      float newFund=fund-money;
      if(newFund<500)
      {
         throw new Exception("Not Sufficient Fund");
      }
      else
      {
         fund=newFund;
         System.out.println("Balance After Withdraw : "+fund);
      }
   }
      public static void main(String arg[])
      {
         Bank b=new Bank();
         b.deposit(1000.00f);
         try
         {
```

```
            float money;
            Scanner sc=new Scanner(System.in);
            System.out.println("Enter Your Amount for withdraw : ");
            money=sc.nextInt();
            System.out.println("Withdrawing amount : "+money);
            b.withdraw(money);
            /* here test with static data so don't worry
            money=300;
            System.out.println("Withdrawing amount : "+money);
            b.withdraw(money); */
          }
          catch(Exception e)
          {
            System.out.println(e.getMessage());
          }

       }
}
```

OUTPUT :

```
javac Bank.java
java Bank

Enter Your Amount for withdraw :
2000
Withdrawing amount : 2000.0
Not Sufficient Fund
```

13. Write a program to create an ArrayList of Products. Traverse the list and print it to the console. Provide a searching of product on name basis using contains() method of List

CODE:

```
import java.io.*;
import java.util.*;
class Product implements Serializable{
 protected int pId;
 protected String pName;
 protected int pRate;
 static int count;
 {
 count++;
 }
 public static void getProductCount(){
 System.out.println(count);
 }
 public String getName(){
 return pName;
 }
```

```
 public void setpName(String pName){
 this.pName= pName;
 }
 public int getpId(){
 return pId;
 }
 public void setpId( int id){
 pId = id;
 }
 public int getpRate(){
 return pRate;
 }
 // return total number of id created
 public int getCount(){
 return count;
 }
 public Product(String name , int rate , int id){
 pName = name;
 pRate = rate;
 pId = id;
 }
 public String toString(){
 return " Product id is " + pId + " name is " +pName ;
 }

 public Product() {
 }
}

public class ArrayList {
 public static void main(String[] args) {
 // Scanner sc = new Scanner(System.in);
 ArrayList list = new ArrayList();
 Product p1 = new Product("RAJ" , 100 , 11);
 Product p2 = new Product("SHYAM", 1000 , 12);
 Product p3 = new Product("RAM" , 100, 13);
 list.add(p1);
 list.add(p2);
 list.add(p3);

 Iterator i = list.iterator();
 while (i.hasNext()) {
 System.out.println(i.next());
 }
 if(list.contains(p1)){
 System.out.println("YES Founded");
 }
 }
```

```
}
```

OUTPUT :

```
Product id is 11 name is RAJ
Product id is 12 name is SHYAM
Product id is 13 name is RAM
YES Founded
```

14.Write a program, to create a TreeSet of Products.Traverse it and provide ordering on base of id.

CODE:

```
import java.io.*;
import java.util.*;
class Product implements Serializable{
 protected int pId;
 protected String pName;
 protected int pRate;
 static int count;
 {
 count++;
 }
 public static void getProductCount(){
 System.out.println(count);
 }
 public String getName(){
 return pName;
 }
 public void setpName(String pName){
 this.pName= pName;
 }
 public int getpId(){
 return pId;
 }
 public void setpId( int id){
 pId = id;
 }
 public int getpRate(){
 return pRate;
 }
 // return total number of id created
 public int getCount(){
 return count;
 }
 public Product(String name , int rate , int id){
 pName = name;
 pRate = rate;
 pId = id;
 }
```

```
 public String toString(){
 return " Product id is " + pId + " name is " +pName ;
 }

 public Product() {
 }
}

public class TreeSet {
 public static void main(String[] args) {
 Comparator cmp = new Comparator() {
 public int compare(Object first , Object second){
 Product pd1 = (Product) first;
 Product pd2 = (Product) second;
 return pd1.getpId() - pd2.getpId();
 }
 };
 Set s = new TreeSet<>(cmp);
 s.add(new Product("B" , 100 , 11));
 s.add(new Product("A", 1000 , 12));
 s.add(new Product("D" , 100, 13));
 s.add(new Product("D" , 100, 13));

 Iterator itr=s.iterator();
 while(itr.hasNext())
 {
 System.out.println(itr.next());
 }
 }
}
```

OUTPUT :

```
 Product id is 11 name is B
Product id is 12 name is A
 Product id is 13 name is D
```

15.Write a program to create a TreeSet of Products. Traverse it and provide ordering on basis of name(Use comparator interface)

CODE:

```
mport java.io.*;
import java.util.*;
class Product implements Serializable{
 protected int pId;
 protected String pName;
 protected int pRate;
 static int count;
```

```
{
count++;
}
public static void getProductCount(){
System.out.println(count);
}
public String getName(){
return pName;
}
public void setpName(String pName){
this.pName= pName;
}
public int getpId(){
return pId;
}
public void setpId( int id){
pId = id;
}
public int getpRate(){
return pRate;
}
// return total number of id created
public int getCount(){
return count;
}
public Product(String name , int rate , int id){
pName = name;
pRate = rate;
pId = id;
}
public String toString(){
return " Product id is " + pId + " name is " +pName ;
}
public Product() {
}
}
public class TreesetName {
public static void main(String[] args) {
Comparator cmp = new Comparator() {
public int compare(Object first , Object second){
Product pd1 = (Product) first;
Product pd2 = (Product) second;
return pd1.getName().charAt(0) - pd2.getName().charAt(0);
}
};
Set s = new TreeSet<>(cmp);
s.add(new Product("RAJ" , 100 , 11));
s.add(new Product("RAM", 1000 , 12));
```

```
s.add(new Product("OM" , 100, 13));
s.add(new Product("OM" , 100, 13));

Iterator itr=s.iterator();
while(itr.hasNext())
{
System.out.println(itr.next());
}
}
}
```

OUTPUT :

```
Product id is 11 name is RAJ
Product id is 12 name is RAM
Product id is 13 name is OM
```

16.Write a program to create a HashSet of Products.Demonstrate that no duplicates value are allowed in HashSet

CODE:

```
import java.io.*;
import java.util.*;
class Product implements Serializable{
 private int pId;
 private String pName;
 private int pRate;
 static int count;
 {
 count++;
 }
 public static void getProductCount(){
 System.out.println(count);
 }
 public String getName(){
 return pName;
 }
 public void setpName(String pName){
 this.pName= pName;
 }
 public int getpId(){
 return pId;
 }
 public void setpId( int id){
 pId = id;
 }
 public int getpRate(){
 return pRate;
 }
 // return total number of id created
```

```
 public int getCount(){
 return count;
 }
 public Product(String name , int rate){
 pName = name;
 pRate = rate;
 }
 public String toString(){
 return " Product id is " + pId + " name is" +pName ;
 }
}
public class HashSetTest {
 public static void main(String[] args)
 {
 HashSet ts1 = new HashSet<Product>();

 Product emp = new Product("ram", 81);
 ts1.add(new Product("A", 10));
 ts1.add(emp);
 ts1.add(new Product("C", 30));
 ts1.add(new Product("B", 20));
 ts1.add(emp);
 Iterator it = ts1.iterator();
 while (it.hasNext())
 {
 Product p = (Product) it.next();
 System.out.println(p.getName() + " has " + p.getpRate());
 }
 }
}
```

OUTPUT :

A has 10
ram has 81
C has 30
B has 20

17.Write a program to demonstrate Thread creation using Runnable interface. While main thread prints 1 to 50 with 1 millisecond pose and child thread print 1 to 100 using 0.5 millisecond pose. Print the name of child and main thread. Main thread needs to wait for child thread to complete

CODE:

```
public class Thread17{

public static void main(String[] args) {
 Runnable r1 = new Runnable() {
 public void run(){
 for (int i = 0; i <= 100; i++) {
```

```
 System.out.println("hello child thread" +i + " the thread address "
+Thread.currentThread().hashCode());
 try{
 Thread.sleep((long) 0.5);
 }
 catch(InterruptedException e){
 System.out.println("Sleep inter");
 }
 }
 }
 };
 Thread t1 = new Thread(r1);
 t1.start();
 for (int i = 0; i <= 50; i++) {
 System.out.println("hello main thread" +i+ "from main thread address " +
Thread.currentThread().hashCode());
 try{
 Thread.sleep(1);
 }
 catch(InterruptedException e){
 System.out.println("Sleep inter");
 }
 }
 t1.interrupt();
 }
}
```

OUTPUT :

```
javac Thread17.java
java Thread17

hello main thread0from main thread address 918221580
hello child thread0 the thread address 2051899962
hello child thread1 the thread address 2051899962
hello child thread2 the thread address 2051899962
hello child thread3 the thread address 2051899962
hello child thread4 the thread address 2051899962
hello child thread5 the thread address 2051899962
hello child thread6 the thread address 2051899962
hello child thread7 the thread address 2051899962
hello child thread8 the thread address 2051899962
hello child thread9 the thread address 2051899962
hello child thread10 the thread address 2051899962
hello child thread11 the thread address 2051899962
hello child thread12 the thread address 2051899962
hello child thread13 the thread address 2051899962
hello child thread14 the thread address 2051899962
hello child thread15 the thread address 2051899962
```

hello child thread16 the thread address 2051899962
hello child thread17 the thread address 2051899962
hello child thread18 the thread address 2051899962
hello child thread19 the thread address 2051899962
hello child thread20 the thread address 2051899962
hello child thread21 the thread address 2051899962
hello child thread22 the thread address 2051899962
hello child thread23 the thread address 2051899962
hello child thread24 the thread address 2051899962
hello main thread1from main thread address 918221580
hello child thread25 the thread address 2051899962
hello child thread26 the thread address 2051899962
hello child thread27 the thread address 2051899962
hello child thread28 the thread address 2051899962
hello child thread29 the thread address 2051899962
hello child thread30 the thread address 2051899962
hello child thread31 the thread address 2051899962
hello child thread32 the thread address 2051899962
hello child thread33 the thread address 2051899962
hello child thread34 the thread address 2051899962
hello child thread35 the thread address 2051899962
hello child thread36 the thread address 2051899962
hello child thread37 the thread address 2051899962
hello child thread38 the thread address 2051899962
hello child thread39 the thread address 2051899962
hello child thread40 the thread address 2051899962
hello child thread41 the thread address 2051899962
hello child thread42 the thread address 2051899962
hello child thread43 the thread address 2051899962
hello child thread44 the thread address 2051899962
hello child thread45 the thread address 2051899962
hello child thread46 the thread address 2051899962
hello child thread47 the thread address 2051899962
hello child thread48 the thread address 2051899962
hello child thread49 the thread address 2051899962
hello child thread50 the thread address 2051899962
hello child thread51 the thread address 2051899962
hello child thread52 the thread address 2051899962
hello child thread53 the thread address 2051899962
hello child thread54 the thread address 2051899962
hello child thread55 the thread address 2051899962
hello child thread56 the thread address 2051899962
hello child thread57 the thread address 2051899962
hello child thread58 the thread address 2051899962
hello child thread59 the thread address 2051899962
hello child thread60 the thread address 2051899962
hello child thread61 the thread address 2051899962
hello child thread62 the thread address 2051899962

hello child thread63 the thread address 2051899962
hello child thread64 the thread address 2051899962
hello child thread65 the thread address 2051899962
hello child thread66 the thread address 2051899962
hello child thread67 the thread address 2051899962
hello child thread68 the thread address 2051899962
hello child thread69 the thread address 2051899962
hello child thread70 the thread address 2051899962
hello main thread2from main thread address 918221580
hello child thread71 the thread address 2051899962
hello child thread72 the thread address 2051899962
hello child thread73 the thread address 2051899962
hello child thread74 the thread address 2051899962
hello child thread75 the thread address 2051899962
hello child thread76 the thread address 2051899962
hello child thread77 the thread address 2051899962
hello child thread78 the thread address 2051899962
hello child thread79 the thread address 2051899962
hello child thread80 the thread address 2051899962
hello child thread81 the thread address 2051899962
hello child thread82 the thread address 2051899962
hello child thread83 the thread address 2051899962
hello child thread84 the thread address 2051899962
hello child thread85 the thread address 2051899962
hello child thread86 the thread address 2051899962
hello child thread87 the thread address 2051899962
hello child thread88 the thread address 2051899962
hello child thread89 the thread address 2051899962
hello child thread90 the thread address 2051899962
hello child thread91 the thread address 2051899962
hello child thread92 the thread address 2051899962
hello child thread93 the thread address 2051899962
hello child thread94 the thread address 2051899962
hello child thread95 the thread address 2051899962
hello child thread96 the thread address 2051899962
hello child thread97 the thread address 2051899962
hello child thread98 the thread address 2051899962
hello child thread99 the thread address 2051899962
hello child thread100 the thread address 2051899962
hello main thread3from main thread address 918221580
hello main thread4from main thread address 918221580
hello main thread5from main thread address 918221580
hello main thread6from main thread address 918221580
hello main thread7from main thread address 918221580
hello main thread8from main thread address 918221580
hello main thread9from main thread address 918221580
hello main thread10from main thread address 918221580
hello main thread11from main thread address 918221580

hello main thread12from main thread address 918221580
hello main thread13from main thread address 918221580
hello main thread14from main thread address 918221580
hello main thread15from main thread address 918221580
hello main thread16from main thread address 918221580
hello main thread17from main thread address 918221580
hello main thread18from main thread address 918221580
hello main thread19from main thread address 918221580
hello main thread20from main thread address 918221580
hello main thread21from main thread address 918221580
hello main thread22from main thread address 918221580
hello main thread23from main thread address 918221580
hello main thread24from main thread address 918221580
hello main thread25from main thread address 918221580
hello main thread26from main thread address 918221580
hello main thread27from main thread address 918221580
hello main thread28from main thread address 918221580
hello main thread29from main thread address 918221580
hello main thread30from main thread address 918221580
hello main thread31from main thread address 918221580
hello main thread32from main thread address 918221580
hello main thread33from main thread address 918221580
hello main thread34from main thread address 918221580
hello main thread35from main thread address 918221580
hello main thread36from main thread address 918221580
hello main thread37from main thread address 918221580
hello main thread38from main thread address 918221580
hello main thread39from main thread address 918221580
hello main thread40from main thread address 918221580
hello main thread41from main thread address 918221580
hello main thread42from main thread address 918221580
hello main thread43from main thread address 918221580
hello main thread44from main thread address 918221580
hello main thread45from main thread address 918221580
hello main thread46from main thread address 918221580
hello main thread47from main thread address 918221580
hello main thread48from main thread address 918221580
hello main thread49from main thread address 918221580
hello main thread50from main thread address 918221580