Java

***Strings***

public class hello {

    public static void main(String[] args)

    {

    //     String name = "sandy";

    //  // it can be done like

    // //   String nam = new String("sandy");

    //     System.out.println(name.length());

    //  String phone = "6784096";

    //  System.out.println(phone.length());

    // strings

//      //--> concatinate

//   String name1 = "sailaja";

//   String name2 = "sandy";

//   String name21 = "trustfated";

//   String name3 = name1 +" and "+ name2+name21;

//   System.out.println(name3);

       // --> charAt(0)

    //    // shows the character at the position

    //    String name = "sandy";

    //    System.out.println(name.charAt(0));

    //           System.out.println(name.charAt(4));

    //   // --> length

    //   String name = "sandy";

    //   System.out.println(name.length());

    //   String oo = "sailaja";

    //   System.out.println(oo.length());

         // replace

        //  String name = "sailaja";

        //  String name3 = name.replace(name , "sandy");

        //  System.out.println(name3);

         // substring

         String name = "sandy and sailaja";

         System.out.println(name.substring(5,9));

         // statrting position of the char from 0 and end position of the char excluding 0

         // this returns the string from the given position to the given position

}

}

# Arrays

public class hello

{

    public static void main(String[] args)

    {

        // arrays non primitive so use new keyword

        // int[] marks = new int[4] ;

        // marks[0] = 34;

        // marks[1] = 45;

        // marks[2] = 56;

        // marks[3] = 67;

        // System.out.println(marks); this won't run because marks wont print

         int[] marks  = {12,23,43,54};

         // can be declared like this also

        System.out.println(marks[0]);

        System.out.println(marks[1]);

        System.out.println(marks[2]);

        System.out.println(marks[3]);

    }

}

Output

12

23

43

54

Another

import java.util.Arrays;

public class hello

{

    public static void main(String [] args)

    {

        // strings

        // // arrays

         boolean[] marks1 = new boolean[3];

         System.out.println(marks1[2]); // we get false because we didnt assign the marks

         //    int[] marks = new int[3]; // if we knew what elements we are gonna add we dont need new keyword

       int[] marks = {97 ,98 ,95};

       // to print length of the array

       System.out.println("the length of the array"+marks.length);// donot add() to length as we did to strings

       // sort // import java.utill.Arrays

       // sort means put the members in ascending order

      Arrays.sort(marks);

      System.out.println("the sorted array is : "+marks[0]);

            System.out.println(marks[1]);

      System.out.println(marks[2]);

    }

}

Output

false

the length of the array3

the sorted array is : 95

97

98

# 2d arrays

import java.util.Arrays;

public class hello

{

    public static void main(String [] args)

    {

     // 2d arrays      0{0,1,2},1{0,1,2}

     int[][] marks = {{75,98,95},{95,95,98}};

     System.out.println("marks"+marks[1][1]);

     // first student's first subject's marks

     //i.e 1st element of 1st array

         System.out.println("marks"+marks[0][2]);

      // casting

        //  implicit casting consversion of the data small  datatype to big

        double price = 100.32;

        double finalprice = price+ 18; //  18 is int whic can be converted into double

        System.out.println(finalprice);

        // java does implicit conversion by itself

        //  //  explicit casting consversion of the data big datatype to small

        int price1 = 100;

        //int finalprice1 = price1+18.98 ; // this shows an error cuz java doesnt do explicit casting by itself

        int finalprice1 = price1 + (int)18.98;

        // a int functin should be used

        System.out.println(finalprice1);

    }

}

Output

marks95

marks95

casting

118.32

118

Some programmes

public class hello

{

    public static void main(String[] args)

    {

        int n = 5;

        for(int i = 1;i<=n;i++)

        {

            for(int j =1;j<=n;j++)

            {

                System.out.print(“\*”);

            }

            System.out.print("\n");

        }

    }

}

Output

1

12

123

1234

12345

2.

// import java.util.Arrays;

public class hello

{

    public static void main(String [] args)

    {

    //  math fnctions

    System.out.println("math function max value"+Math.max(2354,986));

        System.out.println("math function min value"+Math.min(2354,986));

    }

}

Out put

math function max value 2354

math function min value 986

# Scanner

Import java.util.Scanner to scan the input

import java.util.Scanner;

public class hello {

    public static void main(String[] args) {

        System.out.println("enter the value anything ");

        Scanner sc = new Scanner(System.in);

        String name = sc.next();

        int age = sc.nextInt();

        float weight = sc.nextFloat();

        System.out.println(name);

        System.out.println(age);

        System.out.println(weight);

    }

}

import java.util.Scanner;

public class hello {

    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);

        System.out.println("enter the your name : ");

        String name = sc.nextLine();

        System.out.println("enter your age :");

        int age = sc.nextInt();

        System.out.println("enter your weight : ");

        float weight = sc.nextFloat();

        System.out.println("my details::");

        System.out.println(name);

        System.out.println(age);

        System.out.println(weight);

    }

}

Out put

enter the your name :

My name is Sailaja

enter your age :

18

enter your weight :

79.567

my details::

My name is Sailaja

18

79.567

# Conditional statements

import java.util.Scanner;

public class hello {

    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);

        // year leap or not

        System.out.println("this program is to check whether the year is leap or not :");

        System.out.println("enter the your year : ");

        int year = sc.nextInt();

        if (year % 4 == 0)

            System.out.println("leap year");

        else

            System.out.println("not a leap year");

        // age

        System.out.println("this program is to check whether the age is btw 18 to 25 or not :");

        System.out.println("enter your age :");

        int age = sc.nextInt(); // scanning operator //

        if (age >= 18 && age < 25)

            System.out.println("major");

        else

            System.out.println("minor");

    }

}

Output

this program is to check whether the year is leap or not :

enter the your year :

2098

not a leap year

this program is to check whether the age is btw 18 to 25 or not :

enter your age :

18

Major

If you want to print only one statement in if else its not necessary to add blocks

But to print more than one statements you need to add blocks.

import java.util.Scanner;

public class hello {

    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);

        System.out.println("enter the day");

        String day = sc.next();

        switch (day) {

            case "sunday":

                System.out.println("holiday");

                break;

                case "monday":

                System.out.println("get back to work");

                break;

            case "tuesday":

                System.out.println("be motivated");

                break;

            case "wednesday":

                System.out.println("you are mid of the task");

                break;

            case "thursday":

                System.out.println("do not loose motivation");

                break;

            case "friday":

                System.out.println("you are doing well");

                break;

            case "saturday":

                System.out.println("last working day of this week");

                break;

            default:

                System.out.println("enter the name of the day in small case ");

                break;

        }

    }

}

Output

enter the day

sunday

holiday

# loops

import java.util.Scanner;

public class hello {

    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);

     int num ;

        do{

        System.out.println("enter a number > 0 :");

      num = sc.nextInt();

      System.out.println("number: "+num);

     num;

    }

      while(num >= 0); // untill the user enters negative value the loop ends

      System.out.println("this is the end of the loop ..");

    }

}

Output

enter a number > 0 :

1

number: 1

enter a number > 0 :

2

number: 2

enter a number > 0 :

3

number: 3

enter a number > 0 :

4

number: 4

enter a number > 0 :

2

number: 2

enter a number > 0 :

-8

number: -8

this is the end of the loop ..

import java.util.Scanner;

public class hello {

    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);

     String name ;

     int i = 1;

        do{

        System.out.println("enter names:");

      name = sc.next();

      System.out.println(name);

      i++;

    }

    while(i<=5); // untill the user enters negative value the loop ends

      System.out.println("this is the end of the loop ..");

    }

}

Output

sailaja

enter names:

trustfated

trustfated

enter names:

aprilstark

aprilstark

enter names:

zuchi

zuchi

this is the end of the loop ..

# break and continue

break break’s the code at that condition

continue is to skip that condition and continue the rest.

Prime number check;

import java.util.Scanner;

public class hello {

    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);

        int num , count = 0;

        System.out.println("enter the number to check whether the number is prime or not:");

      num = sc.nextInt();

      for(int i = 1 ; i<= num ;i++)

      {

        if(num%i==0)

        {

            count++;

        }

    }

    if(count==2)

       System.out.println("print the number is prime number");

    else

       System.out.println("the number is not prime number");

    }

}

Output

enter the number to check whether the number is prime or not:

5

print the number is prime number

import java.util.Scanner;

public class hello {

    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);

        int num , count = 0;

        System.out.println("enter the number to check whether the number is prime or not:");

      num = sc.nextInt();

      for(int i = 2 ; i< num ;i++)

      {

        if(num%i==0)

            count = 1;

            break;

        }

        if(count==1)

        System.out.println("the number "+num+" is not prime number");

        else

        System.out.println("the number "+num+" is a prime number");

    }

    // if(count==2)

    //    System.out.println("print the number is prime number");

    }

Output

enter the number to check whether the number is prime or not:

46

the number 46 is not prime number

'c:\Users\HP\.redhat'; & 'C:\Program Files\Java\jdk-17\bin\java.exe' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\HP\AppData\Roaming\Code\User\workspaceStorage\141392c15d58c73a8c98573e673c1a60\redhat.java\jdt\_ws\.redhat\_f27ead50\bin' 'hello'

# Exception handling

These are not errors they are execeptions these may cause problems in running the programe

So these are figured out and caught using try and catch.

import java.util.Scanner;

public class hello {

    public static void main(String[] args) {

        try (Scanner sc = new Scanner(System.in)) {

            int num , count = 0;

            System.out.println("enter the number to check whether the number is prime or not:");

     num = sc.nextInt();

     for(int i = 2 ; i< num ;) // if we dont give i++ in java it takes i++ by default

     {

            if(num%i==0)

                count = 1;

                break;

            }

            if(count==1)

            System.out.println("the number "+num+" is not prime number");

            else

            System.out.println("the number "+num+" is a prime number");

        }

    }

    // if(count==2)

    //    System.out.println("print the number is prime number");

    }

As we used in scanner function .

import java.util.Scanner;

public class hello {

    public static void main(String[] args)

    {

        // exception handiling

        int[] marks = {67,98,78};

       try{

           System.out.println(marks[5]); // this shows exception not error

       } catch(Exception exception)

       {

       System.out.println("the exeception is caught");

    }

    System.out.println("these are the marks of students");

Out put

It asks the compiler to try handiling the exception and after observing the error its caught in catch

Where exception is stored in exception class

This process helps to run a secure programme.

# Mini project

**A number guessing game using do while loop and random math function.**

import java.util.Scanner;

public class hello {

     public static void main(String[] args) {

       int mynum = (int)(Math.random()\*10);

      int n;

      try(Scanner sc = new Scanner(System.in))

      {

      do{

      System.out.println("Guess the number between 1-10: ");

      n = sc.nextInt();

      if(n==mynum+5)

         System.out.println("you number is 5 greater than the ahead");

      if(n==mynum)

      {

         System.out.println("you guessed right");

         break;

      }

        }

      while(n>0);

      }

      if(n<=0)

         System.out.println(mynum);

  }

}

# oops

OOPS : ***Object oriented programming structure***

## Classes and objects

Difference in c++ oops and java oops

In C++ oops we can define a function and declare it

In java oops we cannot define only declare a function

// import java.util.Scanner;

class oops{

  String color;

  String type;

  public void write(String x , String y)

    {

     color = x;

     type = y;

    System.out.println((x+y).length());

  }

  }

public class hello {

     public static void main(String[] args)

     {

       oops c = new oops(); // create objects i.e object c is the new object of the class oops

       c.write("blue","use and throw");

  }

}

## Using this # pointers

// import java.util.Scanner;

class oops{

  String color;

  String type;

  public void write()

    {

    System.out.println(this.color);  //    System.out.println(x); is the same   //

    // this.color is which this is pointer.

    // this in java is used to access its public members like colour in its own function

  }

  }

public class hello {

     public static void main(String[] args)

     {

       oops c = new oops(); // create objects i.e object c is the new object of the class oops

       c.color = "BLACK";

       c.write();

  }

}

## Constructors

// import java.util.Scanner;

class Students{

  String name;

  int roll\_no = 51;

  public void details()

    {

    System.out.println(this.name);

        System.out.println(this.roll\_no);  //    System.out.println(x); is the same   //

  //    System.out.println(x); is the same   //

    // this.color is which this is pointer.

    // this in java is used to access its public members like colour in its own function

  }

  // constructors

  Students() {

    System.out.println("this constructor is called ");

  }

  }

public class hello {

     public static void main(String[] args)

     {

       Students s1 = new Students(); // create objects i.e object c is the new object of the class oops

       s1.name = "sailaja";

       s1.details();

  // java calls the constructor by default even whether we give create it or not

  }

}

Output

this constructor is called

sailaja

51

// import java.util.Scanner;

class Students{

  String name;

  int roll\_no = 51;

  public void details()

    {

    System.out.println(this.name);

        System.out.println(this.roll\_no);  //    System.out.println(x); is the same   //

  //    System.out.println(x); is the same   //

    // this.color is which this is pointer.

    // this in java is used to access its public members like colour in its own function

  }

  // constructors parameterized

  Students(String name , int roll\_no) {

    System.out.println("this constructor is called ");

    this.name = name;

    this.roll\_no = roll\_no;

  }

  }

public class hello {

     public static void main(String[] args)

     {

       Students s1 = new Students("sailaja" , 51); // create objects i.e object c is the new object of the class oops

         s1.details();

  // java calls the constructor by default even whether we give create it or not

  }

}

Output

this constructor is called

sailaja

51

class sandy{

  int age ;

  float weight;

  sandy(){

    System.out.println("this is an empty constructor");

  }

  sandy(int a , float w)

  {

    age = a ;

    weight = w;

  }

  public void display()

  {

    System.out.println(age);

    System.out.println(weight);

  }

}

public class hello

{

    public static void main(String [] args)

    {

      sandy s = new sandy(18 , 76.5F);

      s.display();

    }

}

Output

18

76.5

class sandy{

  int age ;

  float weight;

  sandy(){

    System.out.println("this is an empty constructor");

  }

  sandy(int a , float w)

  {

    age = a ;

    weight = w;

  }

  public void display()

  {

    System.out.println(age);

    System.out.println(weight);

  }

}

public class hello

{

    public static void main(String [] args)

    {

      sandy s = new sandy(18 , 76.5F);

      sandy c = new sandy(); // we can call  any number of constructors by creating different objects

      c.display();

      s.display();

    }

Output

0

0.0

18

76.5

## Copy constructors

Copies the properties from other object to another objects

class sandy{

  int age ;

  float weight;

  sandy(){

    System.out.println("this is an empty constructor");

  }

  sandy(int a , float w)

  {

    age = a ;

    weight = w;

  }

  sandy(sandy s2 )

  {

     this.age = s2.age;

     this.weight = s2.weight;

  }

  public void display()

  {

    System.out.println(age);

    System.out.println(weight);

  }

}

public class hello

{

    public static void main(String [] args)

    {

      sandy s1 = new sandy(18 , 76.5F);

      sandy s2 = new sandy();

      sandy s3 = new sandy(s1);

      s3.display();

      // we can copy one object's values to another objects like s1's values are copied to s3 object

      // with the help of copy constructurs

      // we can call  any number of constructors by creating different objects

      s1.display();

      s2.display();

    }

}

// we can copy one object's values to another objects like s1's values are copied to s3 object

      // with the help of copy constructurs

Output

18

76.5

18

76.5

oops

encapsulation

polymorphism

inheritance

abstraction

We donot need destructors in java unlike c++

## polymorphism