JAVA

Format

```
class as{
    Run | Debug
    public static void main(String args[]){
        //code to be executed


    }
}
```

Comments

```
    //  TYPE 1 : MANY LINE


/*
 int x=4;
  for(int i=0;i<10;i++){
    System.out.println(x*i);
  }
*/


    //TYPE 2 : SINGLE LINE


// int x=4;
//   for(int i=0;i<10;i++){
//     System.out.println(x*i);
//   }
```

Data Types and variable

```
double x=10.1;
// where,
// double = data type
//   x    = variable
// 10.1   = double literal
float  t=10.1f;
// where,
// float = data type
//   t   = variable
// 10.1  = float literal
```

```
//RIGHT  WAY  TO WRITE VARIABLE

        // Name is case sensitive
        int x=8;
        int X=9;
        //can start with only alphabet ,$ ,_
        int _t=7;
        String $="money";
        int age=18;

//WRONG WAY  TO WRITE VARIABLE

        //cannot  start with _
        String _="nothing";
        // Should not be a keyword (like void )
        int void=4;
        // White  space is not allowed
        String my name="rahul";
        //must  not start with digit(like 1)
        String 2="bot";
```

Primitive Data Type

```
// Primitive Data Type

    byte b=2;

    short s=2;

     int i=2;

    long l=2;

   double d=2.1;

    float f=2f;

     char c='+';

boolean bool=false;

// String is not a primitive data type. Java.lang package provides the String class therefore, it is an object
type. You can create a string variable directly like any other variables as -
    String S="rahul";
```

JAVA

Scanner Class

```
(import java.util.Scanner)
```

```
    Scanner sc=new Scanner(System.in);

    String s=sc.next();

    char c=sc.next().charAt(index:0);

  //it is needed to put sc.nextline before String sl=sc.nextLine();
    sc.nextLine();//it will consume press(\n) after input like enter 2 and then (press)enter
    String sl=sc.nextLine();

    int x=sc.nextInt();

    float f=sc.nextFloat();

    double d=sc.nextDouble();

    boolean b=sc.nextBoolean();

    short sh=sc.nextShort();

    long l=sc.nextLong();

    byte by=sc.nextByte();
```

Data Type Conversion /widening/implicit conversion

Byte→short→int→ long→ float→double

Type Casting / Narrowing /Explicit Conversion

```
float f=34.56f;
  int x=(int)f;
```

Operators

1. Arithmetic
   a. Binary (mathematical) operator: + - * / %
   b. Unary operator: ++x x++ --x x—
   c. Ternary operator: ?:

```
    String   x        =    21  > 3 ?     "greater" : "smaller";
// datatype  variable  =   condtion ?      true     :    false
```

2. Relational (== != > >= < <=)
3. Logical (&& || !)
4. Bitwise (
   a. &(bitwise and)
   b. |(bitwise or)
   c. <<(shift left)
   d. >>(shift right)
   e. ~(one's complement)
   f. ^(bitwise exclusive or))
5. Assignment (= += -= *= %= /=)

Break and Continue Statement

Break(to exit loop)

```
for(int i=0;i<10;i++){
    if(i==5){
        System.out.println(x:"break");
        break;
    }
    System.out.print(i+" ");
}
System.out.println(x:"understood");
```

Continue(to skip specific condition iteration)

```
for(int i=0;i<10;i++){
    if(i==5){
        System.out.print(s:" here 5 is skip ");
        continue;
    }
    System.out.print(i+" ");
}
System.out.println(x:"understood");
```

Output: break

```
0 1 2 3 4 break
understood
```

Output: continue

```
0 1 2 3 4  here 5 is skip 6 7 8 9 understood
```

Math function

```
Math.max(a:2,b:3);          //3
Math.min(a:2,b:3);          //2
Math.sqrt(a:4);             //4
Math.cbrt(a:27);            //3
Math.random();              //[0-1)
Math.pow(a:2,b:3);          //8
Math.abs(-2);               //2
Math.ceil(a:2.3);           //3
Math.floor(a:2.3);          //2
Math.round(a:2.5);          //3
Math.round(a:2.4);          //2
```

```
int X=30;

Math.toDegrees(X);//0.52 rad
Math.toRadians(X);//1718.873  deg
Math.sin(Math.toRadians(X));
// 0(0) 30(1/2) 45(1/root 2) 60(root3/2) 90(1)
Math.exp(X);
Math.log(X);
Math.log10(X);
```

String Function

```
String str="rahul kumara";
//          01234567891011
str.indexOf(str:"a");//1
str.indexOf(str:"a",fromIndex:3);//9
str.lastIndexOf(str:"a");//11
str.contains(s:"ahul");//true
str.startsWith(prefix:"ra");//true
str.endsWith(suffix:"ra");//true
str.replace(target:"a",replacement:"rt");//rrthul kumrtrrt
String str2="singh";
str.concat(str2);//rahul kumarasingh
```

```
String x="hello";
String a="kumar";

x.compareTo(a);//comparing if same=0,small<0,greater>0 && A!=a

x.compareToIgnoreCase(a);//a=A

x.equals(a);//wheter string  a and x are same

x.length();//number of chaaracter in string

x.charAt(index:2);//help to reach to index of string

x.trim(); //remove extraspace from left and right side of text

x.toUpperCase();//convert whole string to uppercase

x.toLowerCase();//convert whole string to lowercase

x.substring(beginIndex:2,endIndex:3);//between index [2  and 3)
x.substring(beginIndex:2);//between index[2 to whole string]


Character.toUpperCase(codePoint:2);//convert specific index of string to uppercase

String t="23";
int z=Integer.valueOf(t);//convert String to type int

//STRINGBUILDER FUNCTION
StringBuilder sb=new StringBuilder(str:"hello");
sb.append(c:'a');//add 'a' to last of string
Integer aa=10;
aa.toString();//change object to string
```

Conditional Branching/Selectional Control/Decision Making

If Statement

```
if(condition_1){
    //code to be executed

}
```

if else Statement

```
if(condition_1){
    //code to be executed

}
else{
    //else code to be executed

}
```

Else if Statement

```
if(condition_1){
    //code to be executed

}

else if(condition_2){
    //else if code to be executed

}
else if(condition_3){
    //another else if code to be executed

}
else{
    //else code to be executed

}
```

Nested if Statement

```
if(condition_1){
    //code to be executed
    if(condition_2){
        //that code to be executed
    }
    else{
        //else code to be excuted
    }

}
else{
    if(condition_3){
        //that code to be executed
    }
    else{
        //else code to be excuted
    }

}
```

Switch statement

```
int condition =3;
char condition_2='c';

switch(condition){
    case 1://code  1
    break;
    case 2://code  2
    break;
    case 3://code  3
    break;
    default://if no case is matched with condition in switch

}
  switch(condition_2){
    case 'a'://code  1
    break;
    case 'b'://code  2
    break;
    case 'c'://code  3
    break;
    default://if no case is matched with condition in switch

}
```

Loop Statement

1)Exit Controlled Loop/post tested loop (do while loop)

```
do{
 //code to be executed


}while(condition);
```

2)Entry Controlled Loop/pre tested loop (for loop, while loop)

```
for(int i=0;i<10;i++){
 //code to be executed

}
```

```
while(condition){
 //code to be executed


}
```

Function

Here value is passed in function by call by value

```
class as {
    public static void sum(){
        System.out.println(x:"no parameter");
    }
    public static int sum(int x,int y){
        return x+y;
    }
    public static float sum(float x,float y){
        return x+y;
    }
      public static int sum(int x,int y,int z){
        return x+y+z;
    }
    public static boolean sum(int x){
        if (x>0){
            return true;
        }
        return false;
    }
    public static char sum(char c){
        c++;
        return 'a';
    }
    public static String t(String c){
        return  c;
    }

    Run | Debug
    public static void main(String args[]){


    }
}
```

Function overloading:

1. Parameter (same name different parameter)

```
public static int sum(int x,int y){
    return x+y;
}

  public static int sum(int x,int y,int z){
    return x+y+z;
}
```

2. Datatype (same name but parameter datatype different)

```
  public static int sum(int x,int y){
      return x+y;
  }

  public static float sum(float x,float y){
      return x+y;
  }
```

Array

For understanding only

1. *Int dim[row]*
2. *Int dim2[row][column]*
3. *Int dim3[depth][row][column] (generally not used but can be)*

One dimensional Array

```
//one way
int mark[]=new int[2];
mark[0]=76;
mark[1]=89;

//another way
int num[]= {76,78};
// index :  0   1
//memory :1000 1004 (+4 because it is int)
```

Multidimensional Array

2d and 3d array and many more

```
//one way
int mark[][]=new int[2][2];
mark[0][0]=76;
mark[0][1]=89;
mark[1][0]=76;
mark[1][1]=89;

//another way
int num[][]= {{1,2}  ,  {3,4}};
// index :    00 01     10 11
//memory :1000 1004   2000 2004(row wise)
//memory :1000 2000    1004 2004(column wise)
```