Java

With NOYON

CodeMaster Noyon

JAVA: Data Types

In Java Programming language, there are two types of variables: 1. primitive & 2. Non-primitive

Primitive: byte, short, char, Boolean, int, long, float, double.

Non-Primitive: String, Array, Class, Object, Interface.

```
String Name = “Noyon Sarker”;
int age = 24;
int x = 10, y = 20;
float num1 = 39.46f;
double num2 = 50.25d;
```

Note: After every line, you must use ; sign, otherwise you get error. For float must use f and for double must use d.

JAVA: Type Conversion & output

For getting output, you can follow two processes:

1. System.out.print(“Hello world”); [It doesn’t give next-line statement]
2. System.out.println(“Hello world”); [It gives next-line statement]

Type Conversion:

```
int num1 = 10;
float num2 = 20.50f;
```

Convert int to float and float to int. That means short to long and long to short.

```
System.out.println( (float) num1);  [Returns 10.0]
System.out.println( (int) num2);  [Returns 20]
```

JAVA: Input

For taking input from user, we must import one package. Import above the class.

```
Import java.util.*;
```

Into the class, there is some steps;

1. First make a scanner object from scanner class: `Scanner sc = new Scanner(System.in);`
2. Take input and store into the variable: `String name = sc.next();`
3. Display the input variable: `System.out.println(name);`

Note: next() method just take first word. If you want to take a whole sentence as a input then you should use nextLine() method.

```
Scanner sc = new Scanner(System.in);
String name = sc.nextLine();
System.out.println(name);
```

JAVA: Conditional Statement

Conditional statement, check the condition and return output according to the condition.

I. if

II. else

III. else if

Switch case is also use for checking the conditions:

```
int num = 2;
switch (num){

 case 1:
        System.out.println("This is one");
    case 2:
        System.out.println("this is two");

 case 3:
        System.out.println("this is three");
}
```

JAVA: Conditional Statement

Switch case is also use for checking the conditions: break & default

```
int num = 2;
switch (num){
 case 1:
    System.out.println("This is one");
    break;

case 2:
    System.out.println("this is two");
    break;

case 3:
    System.out.println("this is three");
    break;

default:
    System.out.println("Not match");

}
```

JAVA: Loops

In Java, there are three types of loops:

1. For loop [initial state, condition check, update] Initial state is execute just one time
2. While loop
3. Do while loop

For loop:

```
for (int num = 0; num < 10; num++){
    System.out.println(num);
}
```

While loop:

```
int num = 0;
while (num < 5){
    System.out.println(num);
    num++;
}
```

Do while loop:

```
int num = -1;
do{
    System.out.println(num);
    num++;
}while (num > 5);
```

JAVA: Pattern

Pattern 01: Solid Rectangle


```java
int row, col;
row = 4;
col = 5;

//outer loop for -> row
for (int i = 1; i<=row; i++){
    //inner loop for -> col
    for (int j = 1; j<=col; j++){
        System.out.print("*");
    }
    //for new line
    System.out.println();
}
```

Pattern 02:Hollow Rectangle


```java
//outer loop for -> row
for (int i=1; i<=row; i++){
    for (int j=1; j<=col; j++){
        if(i==1 || i==row || j==1 || j==col){
            System.out.print("*");
        }else System.out.print(" ");
    }
    System.out.println();
}
```

JAVA: Pattern

Pattern 03: Half pyramid


```java
int n;
n=10;

//outer loop for -> row
for (int i=1; i<=n; i++){
    for (int j=1; j<=i; j++){
        System.out.print("*");
    }
    System.out.println();
}
```

Pattern 04: Inverted half pyramid


```java
int n;
n=10;
//outer loop for -> row
for (int i=1; i<=n; i++){
    for (int j=n; j>=i; j--){
        System.out.print("*");
    }
    System.out.println();
}
```

JAVA: Pattern

Pattern 05: inverted half pyramid (rotated 180 deg)


```java
int n = 5;
for (int i=1; i<=n; i++){
    for (int j=1; j<=n-i; j++){
        System.out.print(" ");
    }
    for (int k=1; k<=i; k++){
        System.out.print("*");
    }
    System.out.println();
}
```

Pattern 06:


```java
int n = 5;
for (int i=1; i<=n; i++){
    for (int j = 1; j<=i-1; j++){
        System.out.print(" ");
    }
    for (int k=n; k>=i; k--){
        System.out.print("*");
    }
    System.out.println();
}
```

JAVA: Pattern

Pattern 07: Half pyramid with number

```
1
12
123
1234
12345
```

```java
for (int i=1; i<=n; i++){
    for (int j=1; j<=i; j++){
        System.out.print(j);
    }
    System.out.println();
}
```

Pattern 08:inverted half pyramid with numbers

```
12345
1234
123
12
1
```

```java
for (int i=1; i<=n; i++){
    for (int j=1; j<=n-i+1; j++){
        System.out.print(j);
    }
    System.out.println();
}
```

JAVA: Pattern

Pattern 09: Floyd's tringle

```
1
2 3
4 5 6
7 8 9 10
11 12 13 14 15
```

```java
int n, num;
n=5;
num=1;
for (int i=1; i<=n; i++){
    for (int j=1; j<=i; j++){
        System.out.print(num+" ");
        num++;
    }
    System.out.println();
}
```

Pattern 10: 0-1 triangle (calculate the sum of every element's position)

If sum even then 1 printed, if sum odd then 0 printed

```
1
0 1
1 0 1
0 1 0 1
1 0 1 0 1
```

```java
for (int i=1; i<=n; i++){
    for (int j=1; j<=i; j++){
        if((i+j)%2==0){
            System.out.print(1+" ");
        }else System.out.print(0+" ");
    }
    System.out.println();
}
```

JAVA: Complex Pattern

Pattern 11: Butterfly Pattern

```
*      *
**    **
***  ***
********
********
***  ***
**    **
*      *
```

```java
int n = 4;
//upper part
for (int i=1; i<=n; i++) {
    //1st part start
    for (int j = 1; j <= i; j++) {
        System.out.print("*");
    }
    //space print
    for (int j = 1; j <= 2 * (n - i); j++) {
        System.out.print(" ");
    }
    //2nd part
    for (int j = 1; j <= i; j++) {
        System.out.print("*");
    }
    System.out.println();
}
```

```java
//lower half
 for (int i=n; i>=1; i--) {
     //1st part start
     for (int j = 1; j <= i; j++) {
         System.out.print("*");
     }
     //space print
     for (int j = 1; j <= 2 * (n - i); j++) {
         System.out.print(" ");
     }
     //2nd part
     for (int j = 1; j <= i; j++) {
         System.out.print("*");
     }
     System.out.println();
 }
```

Pattern 12: Solid Rhombus

```
    *****
   *****
  *****
 *****
*****
```

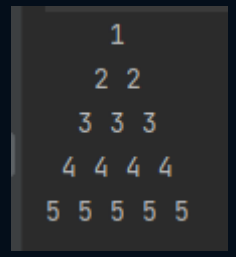
```java
int n = 5;
for (int i=1; i<=n; i++){
    for (int j=1; j<=n-i; j++){
        System.out.print(" ");
    }
    for (int j=1; j<=n; j++){
        System.out.print("*");
    }
    System.out.println();
}
```

JAVA: Complex Pattern

Pattern 13:Holo Rhombus


```
for (int i=1; i<=n; i++){
    for (int j=1; j<=n-i; j++){
        System.out.print(" ");
    }
    for (int j=1; j<=n; j++){
        if(i==1 || i==n || j==1 || j==n){
            System.out.print("*");
        }
        else {
            System.out.print(" ");
        }
    }
    System.out.println();
```

Pattern 14: Number pyramid




```
int n = 5;
for (int i=1; i<=5; i++){
    //for space print
    for (int j=1; j<=n-i; j++){
        System.out.print(" ");
    }
    //for number print
    for (int j=1; j<=i; j++){
        System.out.print(i+" ");
    }
    System.out.println( );
}
```

JAVA: Complex Pattern

Pattern 15: Palindromic

```
    1
   212
  32123
 4321234
543212345
```

```
int n = 5;
for (int i=1; i<=n; i++){
    //space
    for (int j=1; j<=n-i; j++){
        System.out.print(" ");
    }
    for (int j=i; j>=1; j--){
        System.out.print(j);
    }
    if (i>=2){
        for (int j=2; j<=i; j++){
            System.out.print(j);
        }
    }
    System.out.println();
}
```

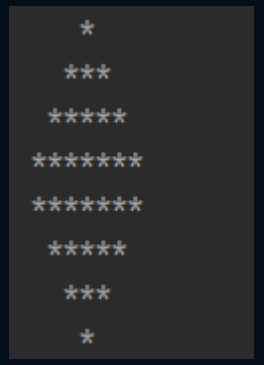
Pattern 16: palindromic -2

```
    11
   2112
  321123
 43211234
5432112345
```

```
int n = 5;
for (int i=1; i<=n; i++){
    //space
    for (int j=1; j<=n-i; j++){
        System.out.print(" ");
    }
    for (int j=i; j>=1; j--){
        System.out.print(j);
    }
    for (int j=1; j<=i; j++){
        System.out.print(j);
    }
    System.out.println();
}
```

JAVA: Complex Pattern

Pattern 15: Diamond pattern




```java
//upper part
for (int i=1; i<=n; i++){
    //space
    for (int j=1; j<=n-i; j++){
        System.out.print(" ");
    }
    for (int j=1; j<=(2*i)-1; j++){
        System.out.print("*");
    }
    System.out.println();
}
```

```java
//lower part
for (int i=n; i>=1; i--){
    //space
    for (int j=1; j<=n-i; j++){
        System.out.print(" ");
    }
    for (int j=1; j<=(2*i)-1; j++){
        System.out.print("*");
    }
    System.out.println();
}
```

JAVA: Function

Function is a block of code. That can be re-use multiple times. Syntax of function:

```
returnType function_name (type para1, type para2) {
    // code
}
```

In Java, all the functions are stored into a stack in the memory.

```
1 usage
public static  void printName(){
    System.out.println("Noyon Sarker");
}
no usages
public static void main(String[] args) {
  printName();
}
```

JAVA: Factorial

Factorial is a mathematical solution. Here,

0! = 1

1! = 1

5! = 5*4*3*2*1

n! = n*(n-1)*(n-2).....1

```
public static int fac(int n){
    if(n<0){
        return 0;
    }else if(n<=1){
        return 1;
    }else {
        return n * fac( n: n-1);
    }
}
```

JAVA: Function

In Java, functions can define the outer of the main function. Some functions return value, some are not:

Return types: int, String, Boolean;

Void type means this function does not return anything.

```
// return the value
public static int addTowNumber(int a, int b){
    return a + b;
}
```

```
// does not return value
public static void printName(String name){
    System.out.println(name);
}
```

JAVA: Array

There are types of array, one dimension and two dimension or 2D array;

// One dimension array

```
int[] arrayName = new int[size];
```

Or

```
int[] arrayName = {2,3,4,5,6,6};
```

Push the value to the array using loop

```
int[]arr = new int[5];
for (int i=0; i<5; i++){
    arr[i] = i;
}
```

JAVA: 2D Array

The syntax of 2D array is:

```
int[][] arr = new int[row][col];
```

Take input from the user for 2D array

```
int[][] arr = new int[2][3];

for (int i=0; i<2; i++){
    for (int j=0; j<3; j++){
        arr[i][j] = sc.nextInt();
    }
}
```