100+ Java Programs with Output

Useful collection of Java Programs

Aniket Pataskar

By : Aniket Pataskar

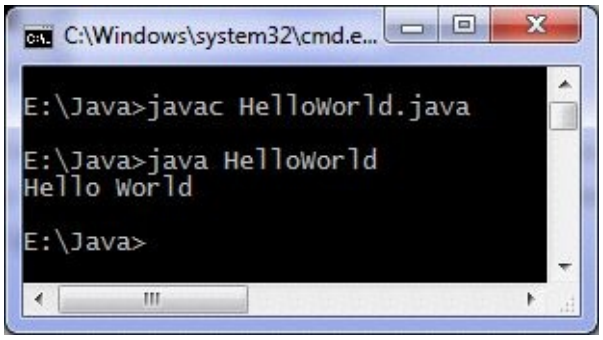
INDEX

1. Hello World example

```
class HelloWorld {
public static void main(String args[]) {
System.out.println("Hello World"); }
}
```

"Hello World" is passed as an argument to println method, you can print whatever you want. There is also a print method which doesn't takes the cursor to beginning of next line as println does. System is a class, out is object of PrintStream class and println is the method.

Output of program:

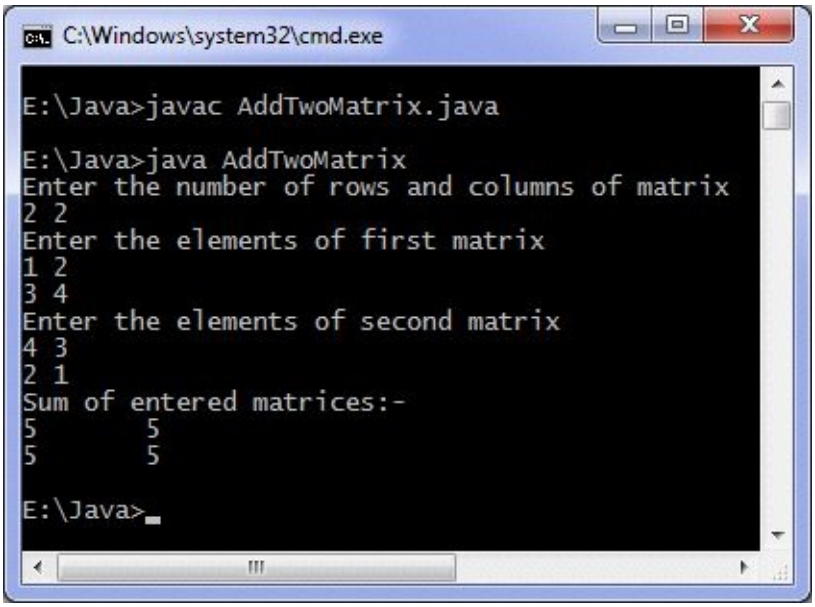


2. Add Two Matrices

```
import java.util.Scanner;
class AddTwoMatrix {
public static void main(String args[]) {
Scanner in = new Scanner(System.in);
System .out.println("Enter
the number of rows and
columns of matrix");
```

```
m = in.nextInt();
n = in.nextInt();
int first[][] = new int[m][n];

int second[][] = new int[m][n]; int sum[][] = new int[m][n];

System .out.println("Enter
the elements of first
matrix");
for( c=0;c<m; c++ )

for ( d = 0 ; d < n ; d++ )
int m, n, c, d;
first[c][d] = in.nextInt();

System .out.println("Enter the elements of second matrix");
for ( c = 0 ; c < m ; c++ )

for ( d = 0 ; d < n ; d++ )
second[c][d] = in.nextInt(); for ( c = 0 ; c < m ; c++ )
for ( d = 0 ; d < n ; d++ )

sum [c][d] = first[c][d] + second[c]
[d]; //replace '+' with '-' to subtract matrices

System .out.println("Sum of entered matrices:-");
for ( c = 0 ; c < m ; c++ )

{
for ( d = 0 ; d < n ; d++ )
System.out.print(sum[c][d] +"\t");

System .out.println();
}
}

}
```

Output of program:



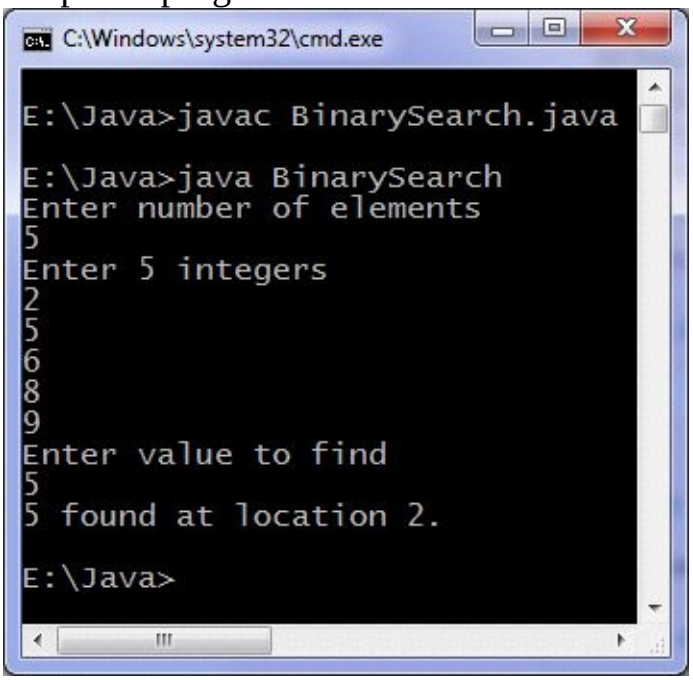
This code adds two matrix, you can modify it to add any number of matrices. You can create a Matrix class and create it's objects and then create an add method which sum the objects, then you can add any number of matrices by repeatedly calling the method using a loop.

3. Binary Search

```
import java.util.Scanner;
class BinarySearch {
public static void main(String args[]) {
int c, first, last, middle, n, search, array[];
Scanner in = new Scanner(System.in);
System.out.println("Enter
number of elements");
n = in.nextInt();
array = new int[n];
System.out.println("Enter "
+ n + " integers");
for (c = 0; c < n; c++) array[c] =
in. nextInt();
System.out.println("Enter
value to find");
search = in.nextInt();
first = 0; last =n-1;
middle = (first + last)/
2;
if ( array[middle] < search ) first = middle + 1; else if
( array[middle] == search ) {
System.out.println(search +
```

```
" found at location " +
(middle + 1) + ".");
break; }
else
last = middle - 1;
middle = (first +
last)/2;
}
if ( first > last )
System.out.println(search + " is not present in the
list.\n");
}
}
while( first <= last ) {
```

Output of program:



Other methods of searching are Linear search and Hashing. There is a binarySearch method in Arrays class which can also be used.

```
import java.util.Arrays;
class BS {
public static void main(String args[])
{
char characters[] =
{ 'a', 'b', 'c', 'd', 'e' };
System.out.println(Arrays.bi
narySearch(characters,
'a'));
```

```
System.out.println(Arrays.bi
narySearch(characters,
'p'));
}

}
```

binarySearch method returns the location if a match occurs otherwise - (x+1) where x is the no. of elements in the array, For example in the second case above when p is not present in characters array the returned value will be -6.

4. Armstrong number

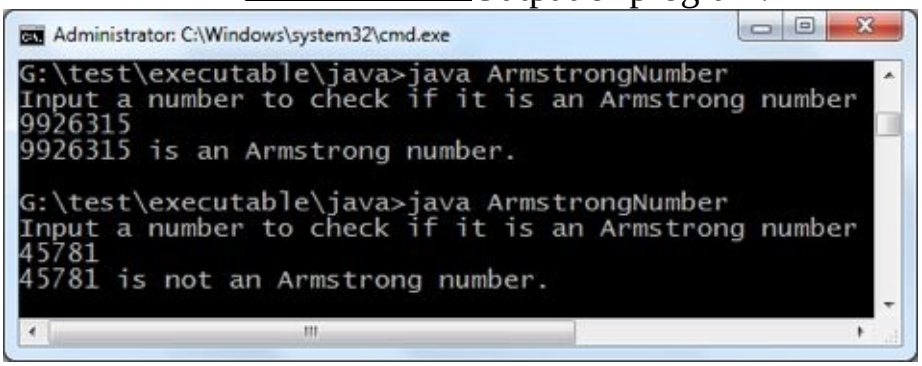
This java program checks if a number is Armstrong or not. Armstrong number is a number which is equal to sum of digits raise to the power total number of digits in the number. Some Armstrong numbers are: 0, 1, 4, 5, 9, 153, 371, 407, 8208 etc.

Java programming code

```
import java.util.Scanner;
class ArmstrongNumber {
public static void main(String args[]) {

int n, sum = 0, temp, remainder, digits = 0;
Scanner in = new Scanner(System.in);

System .out.println("Input a number to check if it is an
Armstrong number"); n = in.nextInt(); temp = n;
// Count number of digits

while (temp != 0) {
%10;
digits++;

temp = temp/10;
}
temp = n;
while (temp != 0) {

remainder = temp
sum = sum + power(remainder, digits);
temp = temp/10;
}
if (n == sum)

System .out.println(n + " is an Armstrong number.");
else

System .out.println(n + " is not an Armstrong number.");
}

static int power(int n, int r) { int c, p = 1;
```

```
for (c = 1; c <= r; c+
+)
return p;
p = p*n; }
}
```

[redacted]Output of program:



Using one more loop in the above code you can generate Armstrong numbers from 1 to n(say) or between two integers (a to b).

5.Bubble sort

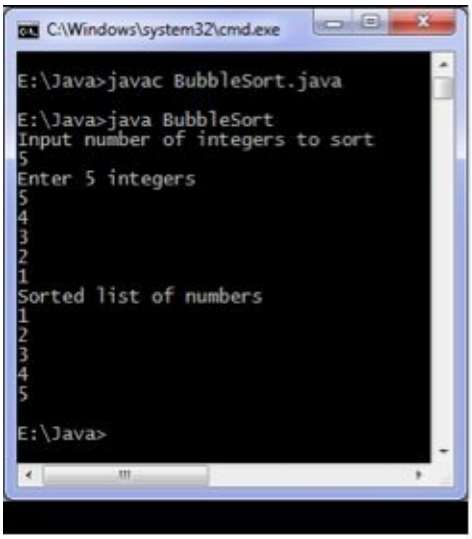
Java program to bubble sort: This code sorts numbers inputted by user using Bubble sort algorithm.

Java programming code

```
import java.util.Scanner;
class BubbleSort { public static void
main(String []args) { int n, c, d, swap; Scanner in = new
Scanner (System.in);
System.out.println("Input
number of integers to
sort");
n = in.nextInt(); int array[] = new
int[n]; System.out.println("Enter " + n + " integers");
for (c = 0; c < n; c++) array[c] = in.nextInt();
for (c = 0; c < ( n - 1 ); c++) {
for (d = 0; d < n - c
- 1; d++) {
if (array[d] >
array[d+1]) /* For descending order use < */
{
```

```
array[d];
swap =
array [d] =
array[d+1];
array[d+1] = swap;
}
} }
System .out.println("Sorted
list of numbers");
for (c = 0; c < n; c++)
System .out.println(array[c])
;
}
}
```

Complexity of bubble sort is $O(n^2)$ which makes it a less frequent option for arranging in sorted order when quantity of numbers is high.
Output of program:

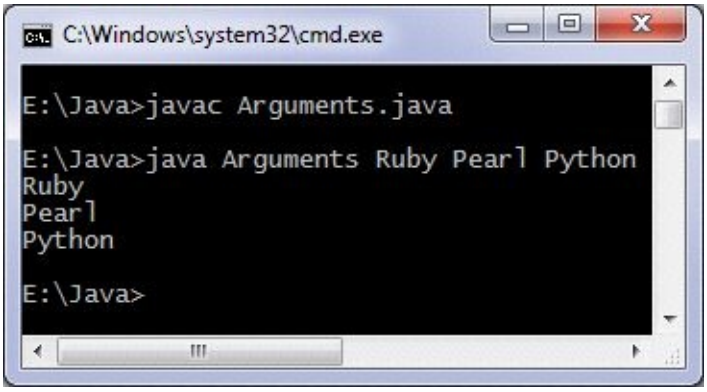


6. Command line arguments

```
class Arguments { public static void main(String[] args) {
for (String t: args) {
System.out.println(t);
}
}
```

```
}
```

Output :



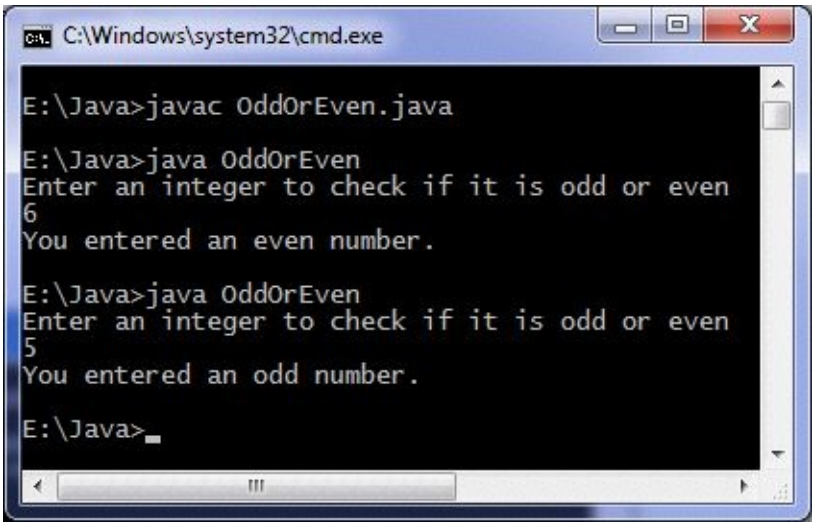
7. Find Odd or Even

This java program finds if a number is odd or even. If the number is divisible by 2 then it will be even, otherwise it is odd. We use modulus operator to find remainder in our program.

Java programming source code

```
import java.util.Scanner;
class OddOrEven {
public static void main(String args[]) {

System .out.println("Enter an
integer to check if it is
odd or even ");
Scanner in = new Scanner(System.in);

x = in.nextInt();
if ( x % 2 == 0 )

System .out.println("You
entered an even number.");
else

System .out.println("You
entered an odd number.");
}

int x;
}
```

Output of program:



Another method to check odd or even, for explanation see:

```
import java.util.Scanner;
class EvenOdd {
public static void main(String args[]) {
int c;

System .out.println(“Input an
integer”);
Scanner in = new Scanner(System.in);

c = in.nextInt(); if ( (c/2)*2 == c )
System.out.println(“Even”); else

System .out.println(“Odd”);
}
}
```

There are other methods for checking odd/even one such method is using bitwise operator.

8. convert Fahrenheit to Celsius

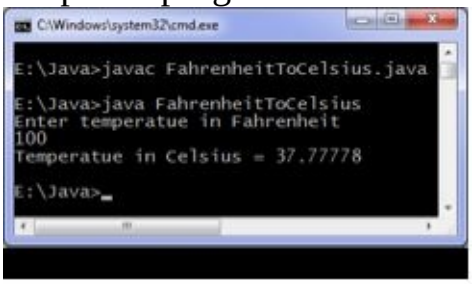
Java program to convert Fahrenheit to Celsius: This code does temperature conversion from Fahrenheit scale to Celsius scale.

Java programming code

```
import java.util.*;

class FahrenheitToCelsius { public static void
main(String[] args) { float temperatue; Scanner in = new

Scanner (System.in);
System.out.println(“Enter
temperatue in Fahrenheit”);

temperatue =
in.nextInt();
temperatue =
```

```
((temperatue - 32)*5)/9;
System.out.println("Temperat
ue in Celsius = " +
temperatue);
}

}
```

Output of program:



For Celsius to Fahrenheit conversion
use
T = 9*T/5 + 32

where T is temperature on Celsius scale. Create and test Fahrenheit to Celsius program yourself for practice.

9. Display Date and Time

Java program to display date and time, print date and time using java program

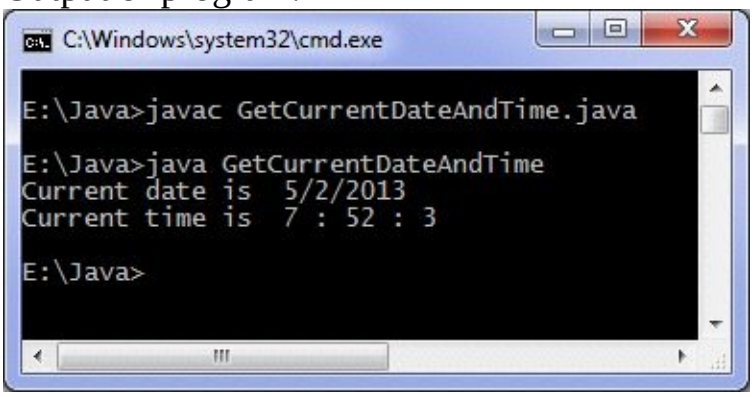
Java date and time program :- Java code to print or display current system date and time. This program prints current date and time. We are using
GregorianCalendar class in our program. Java code to print date and

time is given below :-

```
import java.util.*;
class GetCurrentDateAndTime {
public static void main(String args[]) {
int second, minute, hour;

GregorianCalendar date = new
GregorianCalendar();

day =
date.get(Calendar.DAY_OF_MON
TH);

int day, month, year;

month =
date.get(Calendar.MONTH);
```

```
year =
date.get(Calendar.YEAR);
second =
date.get(Calendar.SECOND);
minute =
date.get(Calendar.MINUTE);
hour =
date.get(Calendar.HOUR); System.out.println("Current date is
"+day+"/"+(month +1)+"/"+year);
System.out.println("Current time is "+hour+" : "+minute +" :
"+second);
}

}
```

Output of program:



Don't use Date and Time class of java.util package as their methods are deprecated means they may not be supported in future versions of JDK. As an alternative of GregorianCalendar class you can use Calendar class.

10. Largest of three integers

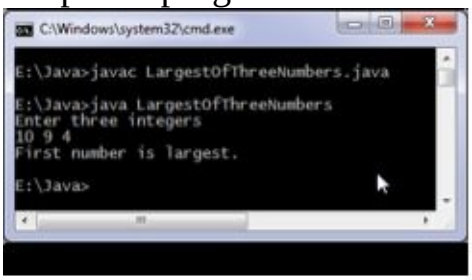
This java program finds largest of three numbers and then prints it. If the entered numbers are unequal then "numbers are not distinct" is printed.

Java programming source code

```
import java.util.Scanner;
class LargestOfThreeNumbers {
public static void main(String args[]) {

System .out.println("Enter
three integers ");
Scanner in = new Scanner(System.in);

x = in.nextInt();
y = in.nextInt();
z = in.nextInt();
```

```
if ( x > y && x > z )
System .out.println("First
number is largest.");
else if ( y > x && y > z)
System .out.println("Second
number is largest.");
int x, y, z;
else if ( z > x && z > y)
System .out.println("Third number is largest.");
else
System .out.println("Entered numbers are not distinct."); }
}
```

Output of program:



11. Java - Sample Programs

Program 1
Find Maximum of 2 nos.

```
class Maxof2{
public static void main(String args[]){ //taking value as command line
argument.
//Converting String format to
Integer value
int i = Integer.parseInt(args[0]); int j = Integer.parseInt(args[1]); if(i > j)
System.out.println(i+" is greater than "+j);
else
System.out.println(j+" is greater
than "+i); }
}
```

Program 2

Find Minimum of 2 nos. using conditional operator

```
class Minof2{
public static void main(String args[]){
```

```
//taking value as command line argument.
//Converting String format to Integer value
int i = Integer.parseInt(args[0]); int j = Integer.parseInt(args[1]); int result = (i<j)?i:j; System.out.println(result+" is a
minimum value"); }
}
```

Program 3

Write a program that will read a float type value from the keyboard and print the following output.

->Small Integer not less than the number.
->Given Number.
->Largest Integer not greater than the number.

```
class ValueFormat
{
public static void main(String args[]){ double i = 34.32; //given number System.out.println("Small Integer
not greater than the number : "+Math.ceil(i));
System.out.println("Given Number : "+i);
System.out.println("Largest Integer not greater than the number : "+Math.floor(i));
```

/*Write a program to generate 5 Random nos. between 1 to 100, and it

should not follow with decimal point.
***/**

```
class RandomDemo{
public static void main(String args[]){
for(int i=1;i<=5;i++){ System.out.println((int)
(Math.random()*100)); }
} }
```

Program 5

/* Write a program to display a greet message according to Marks obtained by student. */

```
class SwitchDemo{
public static void main(String
args[]){
int marks =
Integer.parseInt(args[0]); // take marks as command line argument.
switch(marks/10){ case 10:
case 9:
case 8:
System.out.println("Excellent"); break;
case 7: System.out.println("Very
Good");
break;
case 6:
System.out.println("Good"); break;
case 5: System.out.println("Work
Hard");
break;
case 4:
System.out.println("Poor"); break;
case 3: case 2:
case 1: case 0:
System.out.println("Very
```

```
Poor");
break;

default:
System.out.println("Invalid value Entered"); } }
```

/*Write a program to find SUM AND PRODUCT of a given Digit. */

```
class Sum_Product_ofDigit{

public static void main(String args[]){
int num = Integer.parseInt(args[0]); //taking value as command line argument.

int temp = num,result=0; //Logic for sum of digit while(temp>0){ result = result + temp;
temp—; }
System.out.println("Sum of Digit for "+num+" is : "+result);

//Logic for product of digit temp = num;
result = 1;
while(temp > 0){

result = result * temp;
temp—; }
System.out.println("Product of

Digit for "+num+" is : "+result); }
```

/*Write a program to Find Factorial of Given no. */

```
class Factorial{
public static void main(String args[]){
int num = Integer.parseInt(args[0]); // take argument as command line

int result = 1; while(num>0){
result = result * num;
num—; }
System.out.println("Factorial of Given no. is : "+result); } }
```

Program 8

/*Write a program to Reverse a given no. */

```
class Reverse{

public static void main(String args[]){
int num = Integer.parseInt(args[0]); // take argument as command line

int remainder, result=0; while(num>0){
remainder = num%10;
result = result * 10 + remainder;
num = num/10;

}

System.out.println("Reverse number is : "+result);
} }
```

Program 9

/*Write a program to find Fibonacci series of a given no.

Example : Input - 8
Output - 1 1 2 3 5 8 13 21 */

```
class Fibonacci{
public static void main(String
args[]){
int num =

Integer.parseInt(args[0]); //taking no. as command line argument. System.out.println("*****Fibonac

ci Series*****");
int f1, f2=0, f3=1;

for(int i=1;i<=num;i++){ System.out.print(" "+f3+" "); f1 = f2;  f2 = f3;
```

```
f3 = f1 + f2;
} }
}
```

Program 10
/* Write a program to find sum of all integers greater than 100 and
less than 200 that are divisible by 7 */

```
class SumOfDigit{

public static void main(String args[]){
int result=0;
for(int i=100;i<=200;i++){
if(i%7==0) result+=i;
}
System.out.println("Output of Program is : "+result); } }
```

Program 11
/* Write a program to Concatenate string using for Loop Example: Input - 5
Output - 1 2 3 4 5 */

```
class Join{
public static void main(String
args[]){
int num = Integer.parseInt(args[0]); String result = " "; for(int i=1;i<=num;i++){
result = result + i + " "; }
System.out.println(result); }
}
```

Program 12
/* Program to Display Multiplication Table */

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

int num = Integer.parseInt(args[0]);
System.out.println("*****MULTIPLI CATION TABLE*****"); for(int i=1;i<=num;i++){ for(int j=1;j<=num;j++){
System.out.print(" "+i*j+" "); }
System.out.print("\n"); }
} }
```

Program 13
/* Write a program to Swap the values */

```
class Swap{
public static void main(String args[]){
int num1 = Integer.parseInt(args[0]);
int num2 = Integer.parseInt(args[1]);
System.out.println("\n***Before Swapping***"); System.out.println("Number 1 : "+num1);
System.out.println("Number 2 : "+num2);

//Swap logic
num1 = num1 + num2; num2 = num1 - num2;
num1 = num1 - num2; System.out.println("\n***After
Swapping***"); System.out.println("Number 1 : "+num1); System.out.println("Number 2 :
"+num2);

} }
```

Program 14
/* Write a program to convert given no. of days into months and days.

(Assume that each month is of 30 days)
Example : Input - 69
Output - 69 days = 2 Month and 9 days */

```
class DayMonthDemo{
public static void main(String
args[]){
int num = Integer.parseInt(args[0]); int days = num%30; int month = num/30; System.out.println(num+" days =
"+month+" Month and "+days+" days");
} }
```

Program 15
/*Write a program to generate a Triangle.

eg:
1
22
333
4 4 4 4 and so on as per user given

```
number */ class Triangle{
public static void main(String args[]){
int num = Integer.parseInt(args[0]);

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

Program 16
/* Write a program to Display Invert Triangle.

Example: Input - 5
Output : 55555 4444 333 22
1 */

```
class InvertTriangle{
public static void main(String
args[]){
int num =
Integer.parseInt(args[0]); while(num > 0){
for(int j=1;j<=num;j++){ System.out.print(" "+num+"
");
}
System.out.print("\n");
num—; }
```

/*Write a program to find whether given no. is Armstrong or not.
Example :
Input - 153
Output - 1^3 + 5^3 + 3^3 = 153, so it is Armstrong no. */ class Armstrong{

```
public static void main(String args[]){
int num = Integer.parseInt(args[0]);
int n = num; //use to check at last time
int check=0,remainder; while(num > 0){
remainder = num % 10;
check = check + (int)Math.pow(remainder,3);
num = num / 10; }
if(check == n) System.out.println(n+" is an
Armstrong Number"); else
System.out.println(n+" is not a Armstrong Number");
```

/* Write a program to Find whether number is Prime or Not. */ class PrimeNo{

public static void main(String args[]){
int num = Integer.parseInt(args[0]);

int flag=0;
for(int i=2;i<num;i++){

if(num%i==0) {
System.out.println(num+" is not a Prime Number");
flag = 1;
break; }
} if(flag==0)
System.out.println(num+" is a Prime Number");
} }

Program 19
/* Write a program to find whether no. is palindrome or not. Example :
Input - 12521 is a palindrome
no.
Input - 12345 is not a
palindrome no. */

class Palindrome{
public static void main(String
args[]){
int num =

Integer.parseInt(args[0]);
int n = num; //used at last time
check
int reverse=0,remainder;
while(num > 0){
remainder = num % 10; reverse = reverse * 10 +
remainder;
num = num / 10;
}
if(reverse == n)
System.out.println(n+" is a

Palindrome Number"); else
System.out.println(n+" is not a Palindrome Number"); } }

Core Java Programs [PAGE 3]

Some Java programs which help lot of java beginners to understand the basic fundamentals in Java programming. Most of these programs take input from the command line. *Ex - int num = Integer.parseInt(args[0]);*

/* switch case demo
Example : Input - 124
Output - One Two Four */

class SwitchCaseDemo{
public static void main(String

args[]){ try{
int num = Integer.parseInt(args[0]); int n = num; //used at last time check int reverse=0,remainder; while(num > 0){
remainder = num % 10;
reverse = reverse * 10 + remainder; num = num / 10; }
String result=""; //contains the actual output while(reverse > 0){

“Zero “;
“One “;
“Two “;
“Three “;
“Four “;
“Five “;
“;
“Seven “;
“Eight “;

remainder = reverse % 10; reverse = reverse / 10; switch(remainder){
case 0 :
result = result +

break; case 1 :
result = result +
break; case 2 :
result = result +
break; case 3 :
result = result +
break; case 4 :
result = result +
break; case 5 :
result = result +
break; case 6 :
result = result + “Six
break; case 7 :
result = result +
break; case 8 :
result = result +
break; case 9 :
result = result +
break; default:
result=””; }

}
System.out.println(result);

}catch(Exception e){ System.out.println(“Invalid
Number Format”); }
} }

Program 21
/* Write a program to generate Harmonic Series. Example : Input - 5
Output - 1 + 1/2 + 1/3 + 1/4 + 1/5 = 2.28 (Approximately) */ class HarmonicSeries{
public static void main(String args[]){
int num = Integer.parseInt(args[0]); double result = 0.0; while(num > 0){

result = result + (double) 1 / num;
num—; }
System.out.println(“Output of Harmonic Series is “+result); “Nine “;

Program 22
/*Write a program to find average of consecutive N Odd no. and Even no. */
class EvenOdd_Avg{

public static void main(String args[]){
int n = Integer.parseInt(args[0]);
int cntEven=0,cntOdd=0,sumEven=0,sum Odd=0;
while(n > 0){ if(n%2==0){
cntEven++;
sumEven = sumEven + n; }

else{

```
cntOdd++;
sumOdd = sumOdd + n;
}
n—; }
int evenAvg,oddAvg;
evenAvg = sumEven/cntEven; oddAvg = sumOdd/cntOdd; System.out.println(“Average of first
N Even no is “+evenAvg); System.out.println(“Average of first N Odd no is “+oddAvg);
} }
```

Program 23

```
/* Display Triangle as follow : BREAK DEMO.

1
23
456
7 8 9 10 … N */
class Output1{
public static void main(String
args[]){
int c=0;
int n = Integer.parseInt(args[0]); loop1: for(int i=1;i<=n;i++){ loop2: for(int j=1;j<=i;j++){
if(c!=n){ c++;
System.out.print(c+”
“);
}
else
break loop1;
}
System.out.print(“\n”); }
```

Program 24

```
/* Display Triangle as follow
0
10
101
0 1 0 1 */
class Output2{
public static void main(String
args[]){
for(int i=1;i<=4;i++){
for(int j=1;j<=i;j++){ System.out.print(((i +j)%2)+” “);
}
System.out.print(“\n”);
} }
}
```

Program 25

```
/* Display Triangle as follow
1
24
369
4 8 12 16 … N (indicates no. of
Rows) */
class Output3{
public static void main(String
```

```
args[]){
int n = Integer.parseInt(args[0]);
for(int i=1;i<=n;i++){ for(int j=1;j<=i;j++){
“);
}
} }
System.out.print((i*j)+” System.out.print(“\n”); }
```

12.Java Programs part 2

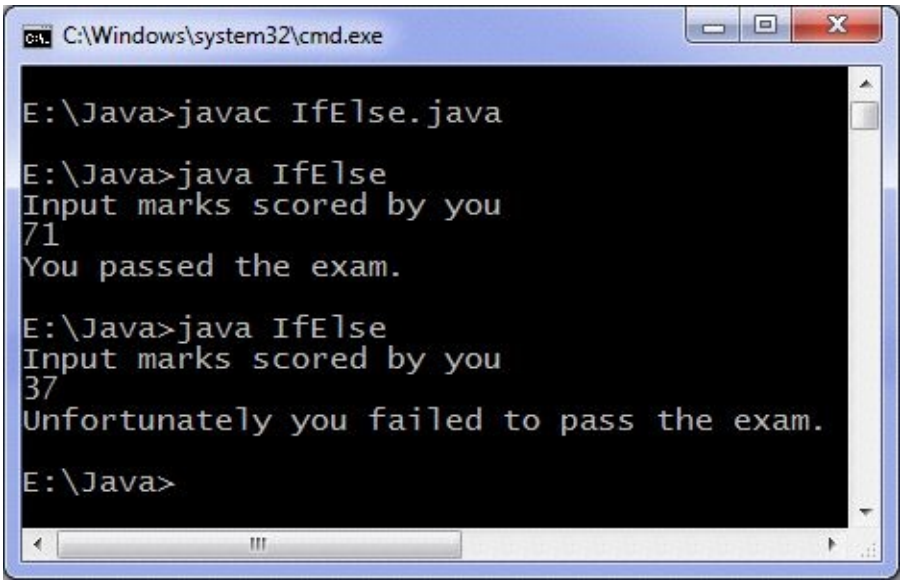
1. Java if else program

Java if else program uses if else to execute statement(s) when a condition is fulfilled. Below is a simple program which explains the usage of if else in java programming language.

Java programming if else
statement

```
// If else in Java code
import java.util.Scanner;

class IfElse { public static void main(String[] args) { int
marksObtained,

passingMarks ;
passingMarks = 40;
Scanner input = new Scanner(System.in);

System.out.println(“Input
marks scored by you”);
marksObtained =

input. nextInt();
if (marksObtained >=
passingMarks) {
System.out.println(“You
passed the exam.”);
}
else {

System .out.println(“Unfortun
ately you failed to pass the
exam.”);
} }

}
```

Output of program:



Above program ask the user to enter marks obtained in exam and the input marks are compared against minimum passing marks. Appropriate message is printed on screen based on whether user passed the exam or not. In the above code both if and else block contain only one statement but we can execute as many statements as

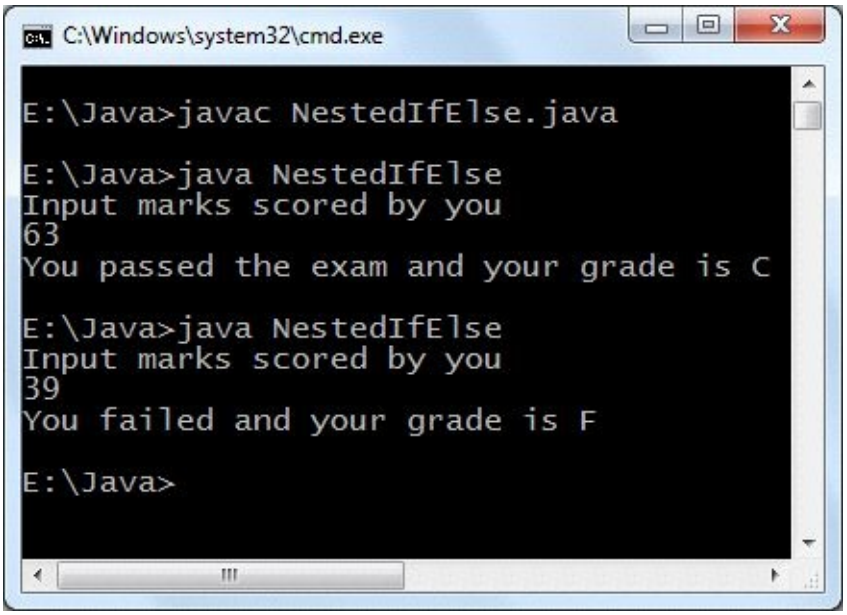
required.

2. Nested If Else statements

You can use nested if else which means that you can use if else statements in any if or else block.

```
import java.util.Scanner;
class NestedIfElse { public static void main(String[] args)
{ int marksObtained, passingMarks; char grade;
passingMarks = 40; Scanner input = new

Scanner (System.in);
System.out.println("Input
marks scored by you");

marksObtained =
input.nextInt();
if (marksObtained >= passingMarks) { if (marksObtained > 90)

grade = 'A';
else if (marksObtained > 75)
else if (marksObtained > 60)
else
grade = 'B';

grade = 'C';
grade = 'D';
System.out.println("You passed the exam and your grade is "
+ grade);
}
else {

grade = 'F';

System .out.println("You failed and your grade is " +
grade);
} }

}
```

Output of program:

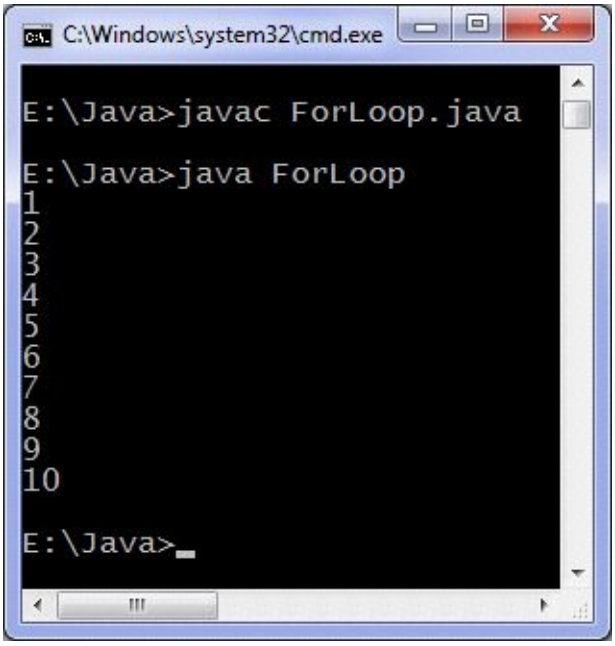

C:\Windows\system32\cmd.exe
E:\Java>javac NestedIfElse.java
E:\Java>java NestedIfElse
Input marks scored by you
63
You passed the exam and your grade is C
E:\Java>java NestedIfElse
Input marks scored by you
39
You failed and your grade is F
E:\Java>

3. Java for loop

Simple for loop example in Java
Example program below uses for loop to print first 10 natural numbers i.e. from 1 to 10.

```
//Java for loop program
class ForLoop { public static void main(String[] args) { int
c;
for (c = 1; c <= 10; c+ +) {
System.out.println(c);

}
}
```

Output of program:



4. Java for loop example to print stars in console

Following star pattern is printed
*

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

```
class Stars {
public static void
main(String[] args) {
int row, numberOfStars;
for (row = 1; row <= 10; row++) { for(numberOfStars = 1;
numberOfStars <= row;

numberOfStars ++) {
System.out.print("*");
}

System.out.println(); // Go to next line } }
```

Above program uses nested for loops (for loop inside a for loop) to print stars. You can also use spaces to create another pattern, It is left for

you as an exercise.
Output of program:

5. Java while loop

Java while loop is used to execute statement(s) until a condition holds true. In this tutorial we will learn looping using Java while loop examples. First of all lets discuss while loop syntax:

while (condition(s)) {
// Body of loop
}

1. If the condition holds true then the body of loop is executed, after execution of loop body condition is tested again and if the condition is

true then body of loop is executed again and the process repeats until condition becomes false. Condition is always evaluated to true or false and if it is a constant, For example while (c) { …} where c is a constant then any non zero value of c is considered true and zero is considered false.

2. You can test multiple conditions such as

```
while ( a > b && c != 0) { // Loop body }
```

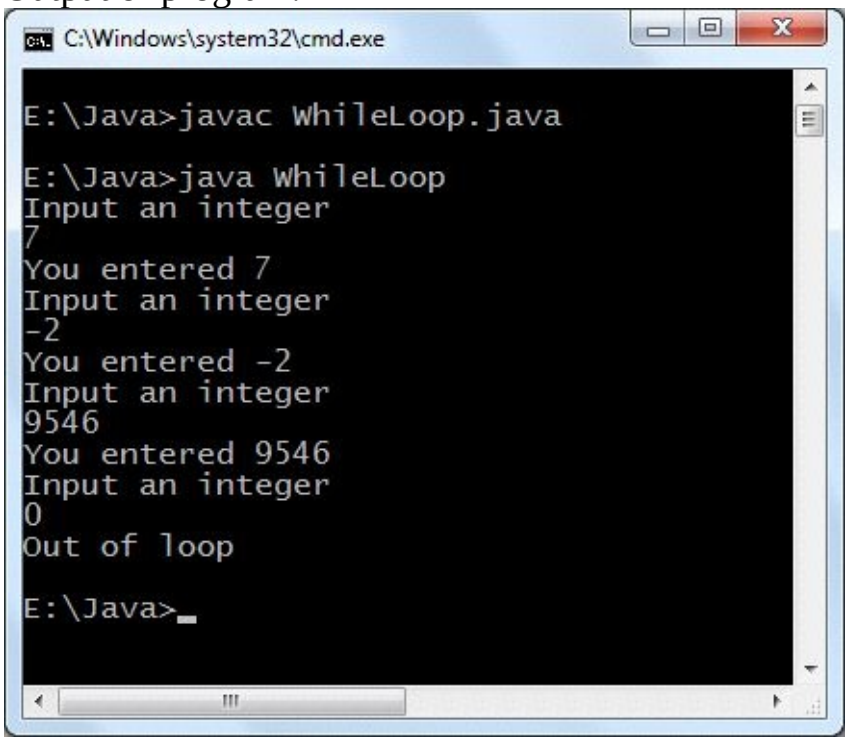
Loop body is executed till value of a is greater than value of b and c is not equal to zero.

3. Body of loop can contain more than one statement. For multiple statements you need to place them in a block using {} and if body of loop contain only single statement you can optionally use {}. It is recommended to use braces always to make your program easily readable and understandable.

Java while loop example
Following program asks the user to
input an integer and prints it until user enter 0 (zero).

```
import java.util.Scanner;
class WhileLoop { public static void main(String[] args) {
int n;
Scanner input = new Scanner(System.in);

System .out.println(“Input an
integer”);
while ((n = input.nextInt()) != 0) {

System .out.println(“You
entered ” + n);
System.out.println(“Input an
integer”);
}

System .out.println(“Out of loop”);
}

}
```

Output of program:



6. Java program to print alphabets

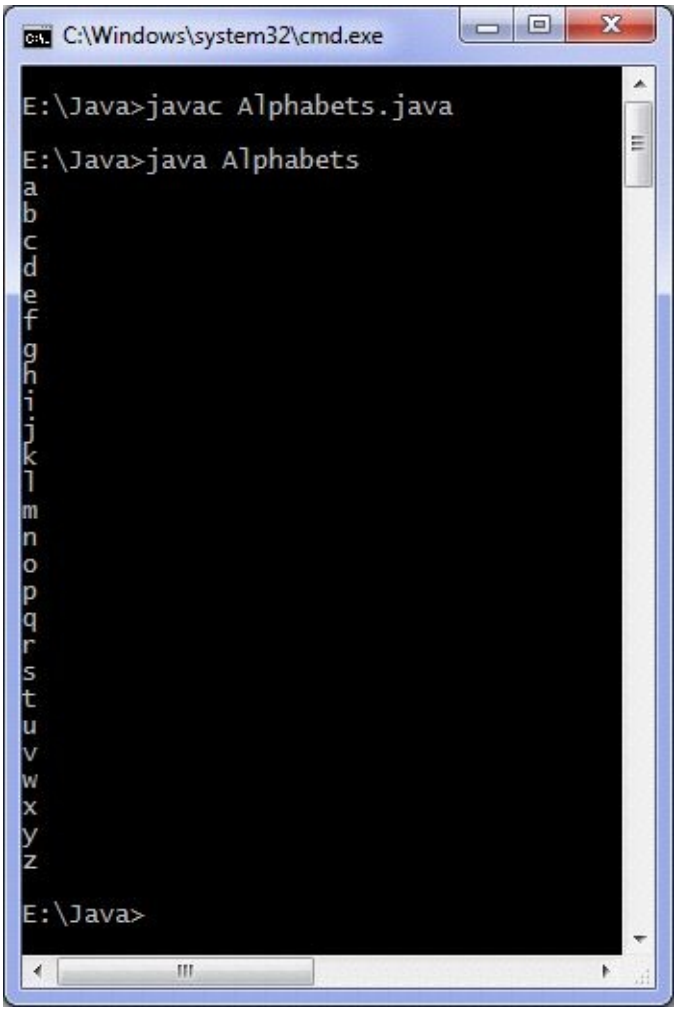
This program print alphabets on screen i.e a, b, c, …, z. Here we print alphabets in lower case.

Java source code

```
class Alphabets {
public static void main(String args[]) {
'z' ; ch++ )
System.out.println(ch);
char ch;
for( ch = 'a' ; ch <=
}
}
```

You can easily modify the above java program to print alphabets in upper case.

Output of program:



Printing alphabets using while loop

(only body of main method is shown): `char c = 'a'; while (c <= 'z') {`

```
[redacted]}
System.out.println(c);
c++;
```

Using do while loop:

```
char c = 'A';
do { System.out.println(c); c++; } while (c <= 'Z');
```

Java program to print multiplication table

This java program prints multiplication table of a number

entered by the user using a
for
loop.

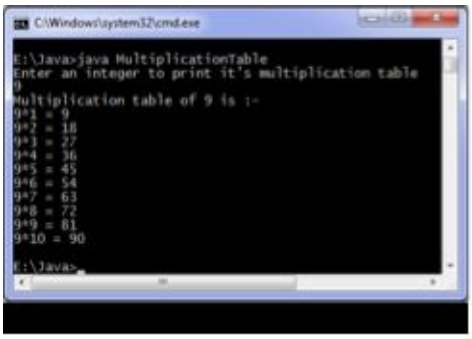
You can modify it for
while
loop for practice.
Java programming source code

```
import java.util.Scanner;
class MultiplicationTable {
public static void main(String args[]) {
int n, c; System.out.println("Enter an
```

while
or
do

```
integer to print it's
multiplication table");
Scanner in = new Scanner(System.in);

n = in.nextInt();
System.out.println("Multipli
cation table of "+n+"
is :-");
for ( c = 1 ; c <= 10 ; c++ )

System.out.println(n+"*"+c+"
= "+(n*c));
}
}
```

Output of program:


C:\Windows\system32\cmd.exe
E:\Java>java MultiplicationTable
Enter an integer to print it's multiplication table
9
Multiplication table of 9 is :-
9*1 = 9
9*2 = 18
9*3 = 27
9*4 = 36
9*5 = 45
9*6 = 54
9*7 = 63
9*8 = 72
9*9 = 81
9*10 = 90
E:\Java>

How to get input from user in java

This program tells you how to get input from user in a java program. We are using Scanner class to get input from user. This program firstly asks the user to enter a string and then the string is printed, then an integer and entered integer is also printed and finally a float and it is also printed on the screen. Scanner class is present in java.util package so we import this package in our program. We first create an object of Scanner class and then we use the methods of Scanner class. Consider the statement
Scanner a = new Scanner(System.in); Here Scanner is the class name, a is the name of object, new keyword is used to allocate the memory and System.in is the input stream. Following methods of Scanner class

are used in the program below :-
1) nextInt to input an integer
2) nextFloat to input a float
3) nextLine to input a string
Java programming source code

```
import java.util.Scanner;
class GetInputFromUser {
public static void main(String args[]) {
Scanner in = new Scanner(System.in);

System.out.println("Enter a
string");
s = in.nextLine();

System .out.println("You
entered string "+s);
System.out.println("Enter an
integer");

a = in.nextInt();
System.out.println("You
entered integer "+a);
System.out.println("Enter a
float");

b = in.nextFloat();
int a; float b; String s;

System .out.println("You
entered float "+b);
}

}
```

Output of program:

There are other classes which can be used for getting input from user and you can also take input from other devices.

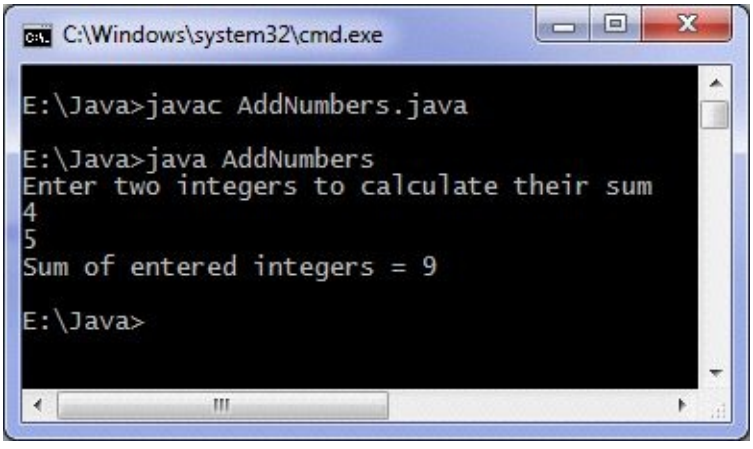
Java program to add two numbers

Java program to add two numbers :-
Given below is the code of java program that adds two numbers which are entered by the user.

Java programming source code

```
import java.util.Scanner;
class AddNumbers {
public static void main(String args[]) {

System .out.println(“Enter
two integers to calculate
their sum “);
Scanner in = new Scanner(System.in);

x = in.nextInt();
y = in.nextInt();
z = x + y;

System.out.println(“Sum of
entered integers = “+z);

}
}
int x, y, z;
```

Output of program:



Above code can add only numbers in range of integers(4 bytes), if you wish to add very large numbers then you can use BigInteger class. Code to add very large numbers:

```
import java.util.Scanner; import
java.math.BigInteger;
class AddingLargeNumbers { public static void
main(String[] args) {

String number1, number2; Scanner in = new
Scanner(System.in);
System.out.println(“Enter
```

```
first large number”);

number1 = in.nextLine();
System.out.println(“Enter
second large number”);

number2 = in.nextLine();
BigInteger first = new
BigInteger(number1);

BigInteger second = new
BigInteger(number2);

BigInteger sum;
sum = first.add(second);
System.out.println(“Result
of addition = ” + sum);
}

}
```

In our code we create two objects of BigInteger class in java.math package. Input should be digit strings otherwise an exception will be raised, also you cannot simply use ‘+’ operator to add objects of BigInteger class, you have to use add method for addition of two objects.
Output of program:

```
Enter first large number 11111111111111
Enter second large number 99999999999999
Result of addition = 111111111111110
```

Java Method example program

```
class Methods {
// Constructor method Methods() {
System .out.println("Construc tor method is called when an
object of it's class is
created");
}
// Main method where
program execution begins
public static void main(String[] args) {
staticMethod();
Methods object = new Methods();
object. nonStaticMethod(); }
// Static method
static void staticMethod() {
System .out.println("Static method can be called without
creating object");
}
// Non static method
void nonStaticMethod() { System.out.println("Non
static method must be called by creating an object"); }
}
```

Output of program:

Java String methods

String class contains methods which are useful for performing operations on String(s). Below program illustrate how to use inbuilt methods of String class.

Java string class program

```
class StringMethods {
public static void main(String args[]) {
int n;

String s = “Java
programming”, t = ””, u =
””;
System.out.println(s); // Find length of string n =
s.length();

System.out.println(“Number
of characters = ” + n);

// Replace characters in
string
t = s.replace(“Java”, “C
++”);

System .out.println(s); System.out.println(t); //
Concatenating string

with another string
u = s.concat(” is fun”); System.out.println(s);
System.out.println(u);

}
}
```

Output of program:

13. Java Programs part 3

Java static block program

Java programming language offers a block known as static which is executed before main method executes. Below is the simplest example to understand functioning of static block later we see a practical use of static block.

Java static block program

```
class StaticBlock { public static void
main(String[] args) {
System.out.println("Main
method is executed.");
}
static {
System .out.println("Static
block is executed before
main method.");
}
}
```

Output of program:

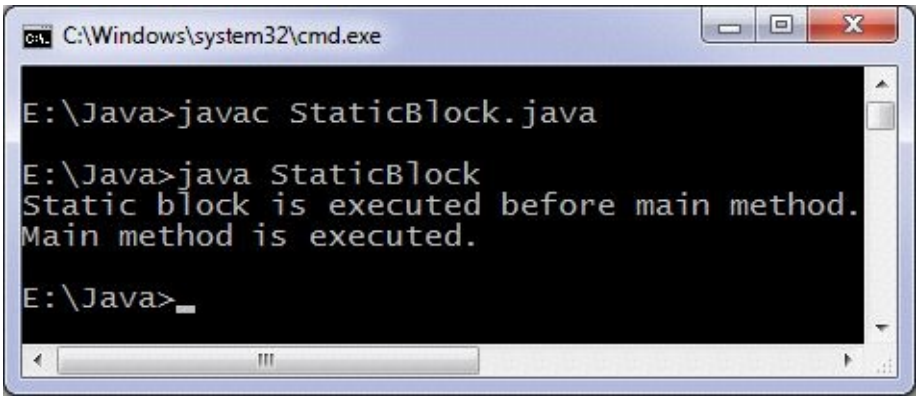
Static block can be used to check conditions before execution of main begin, Suppose we have developed an application which runs only on Windows operating system then we need to check what operating system is installed on user machine. In our java code we check what operating system user is using if user is using operating system other than "Windows" then the program terminates.

```
class StaticBlock { public static void main(String[] args) {
System .out.println("You
are using Windows_NT
operating system.");
}
static { String os =
System.getenv("OS"); if
(os.equals("Windows_NT") !=
```

```
true) { System.exit(1);
} }
}
```

We are using getenv method of System class which returns value of environment variable name of which is passed an as argument to it. Windows_NT is a family of operating systems which includes Windows XP, Vista, 7, 8 and others.

Output of program on Windows 7:



Java static method

Java static method program: static methods in Java can be called without creating an object of class. Have you noticed why we write static keyword when defining main it's because

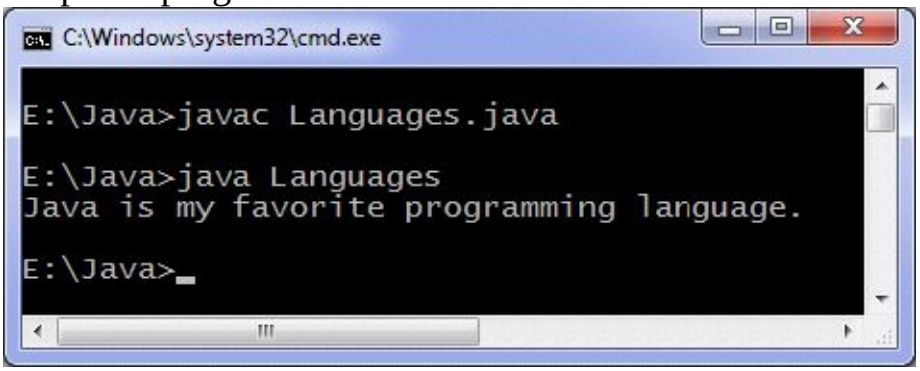
program execution begins from main

and no object has been created yet. Consider the example below to improve your understanding of static methods.

Java static method example
program

```
class Languages { public static void
main (String[] args) {
display();
}
static void display()
{ System.out.println("Java
is my favorite programming
language.");
}
}
```

Output of program:



Java static method vs instance method
Instance method requires an object of
its class to be created before it can be

called while static method doesn't require object creation.

```
class Difference {
public static void main(String[] args) { display();
//calling without object Difference t = new Difference();
t.show(); //calling using object
}
static void display() {
```

```
System .out.println("Programm
ing is amazing.");
}
void show(){ System.out.println("Java
is awesome." );
}
}
```

Output of code:

Using static method of another classes

If you wish to call static method of another class then you have to write class name while calling static method as shown in example below.

```
import java.lang.Math;
class Another { public static void main(String[] args) { int
result;
result = Math.min(10, 20); //calling static method min by
writing class name
System .out.println(result);
System.out.println(Math.max(
100, 200));
}
}
```

Output of program:

```
10
200
```

Here we are using min and max methods of Math class, min returns minimum of two integers and max

returns maximum of two integers.
Following will produce an error:

```
min();
```

We need to write class name because many classes may have a method with same name which we are calling.

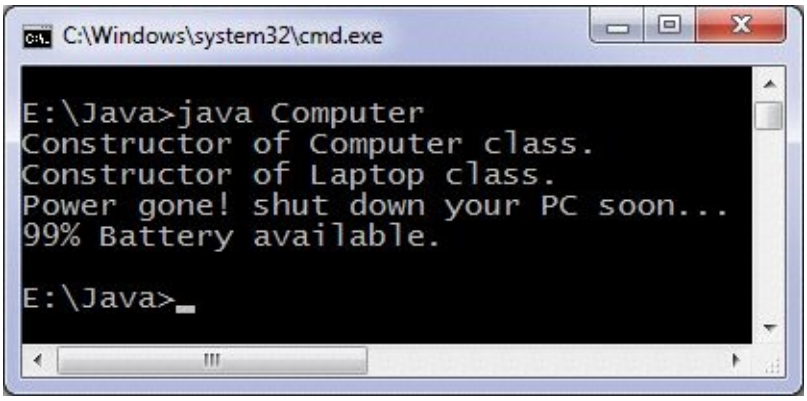
Using multiple classes in Java program

ava program can contain more than one i.e. multiple classes. Following example Java program contain two classes: Computer and Laptop. Both classes have their own constructors and a method. In main method we create object of two classes and call their methods.

Using two classes in Java program

```
class Computer { Computer() {
System .out.println(“Construc
tor of Computer class.”);
}
void computer_method() {
System .out.println(“Power
gone! Shut down your PC
soon…”);
}
public static void main(String[] args) { Computer my = new
Computer();
Laptop your = new
Laptop();
my.computer_method(); your.laptop_method();
}
}
class Laptop {
Laptop() {
System .out.println(“Construc tor of Laptop class.”); }
void laptop_method()
{ System.out.println(“99%
Battery available.” );
}
}
```

Output of program:


C:\Windows\system32\cmd.exe
E:\Java>java Computer
Constructor of Computer class.
Constructor of Laptop class.
Power gone! shut down your PC soon...
99% Battery available.
E:\Java>

Java constructor tutorial with code examples

Constructor java tutorial: Java constructors are the methods which are used to initialize objects.
Constructor method has the same name as that of class, they are called or invoked when an object of class is created and can't be called explicitly. Attributes of an object may be available when creating objects if no attribute is available then default constructor is called, also some of the attributes may be known initially. It is optional to write constructor method in a class but due to their utility they are used.

Java constructor example

```
class Programming { //constructor method Programming() {
System .out.println("Construc
tor method called.");
}
public static void main(String[] args) { Programming object
= new
Programming(); //creating object
}
}
```

Output of program:

This code is the simplest example of constructor, we create class Programming and create an object, constructor is called when object is created. As you can see in output "Constructor method called." is printed.

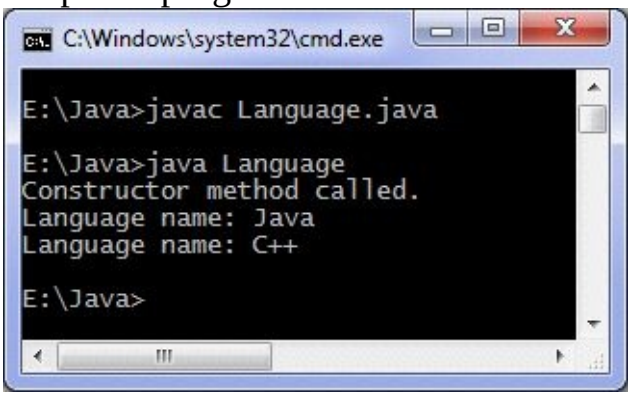
Java constructor overloading

Like other methods in java constructor can be overloaded i.e. we can create as many constructors in our class as desired. Number of constructors depends on the information about attributes of an object we have while creating objects. See constructor overloading example:

```
class Language { String name;
Language() {
System .out.println("Construc
```

```
tor method called.");
}

Language(String t) {
name = t;

}
public static void main(String[] args) { Language cpp = new
Language();
Language java = new Language("Java");

cpp. setName("C++");
java.getName();
cpp.getName();

}
void setName(String t) { name = t; }
void getName() {

System .out.println("Language name: " + name);
}

}
```

Output of program:



When cpp object is created default constructor is called and when java object is created *constructor with argument* is called, setName method is used to set 'name' attribute of language, getName method prints language name.

Java constructor chaining

Constructor chaining occurs when a class inherits another class i.e. in inheritance, as in inheritance sub class inherits the properties of super class. Both the super and sub class may have constructor methods, when an object of sub class is created it's constructor is invoked it initializes sub class attributes, now super class constructor needs to be invoked, to

achieve this java provides a

super

keyword through which we can pass arguments to super class constructor. For more understanding see constructor chaining example:

```
class GrandParent { int a;
GrandParent(int a) { this.a = a; } }
class Parent extends GrandParent { int b;

Parent(int a, int b) { super(a);
this.b = b;
}
void show()
{ System.out.println("GrandPar
ent's a = " + a);
System.out.println("Parent's
b

} }
= " + b);

class Child {
public static void
main(String[] args) { Parent object = new

Parent (8, 9);
object.show();
}

}
```

Output of program:

Constructor method doesn't specify a return type, they return instance of class itself.

Java program to swap two numbers

This java program swaps two numbers using a temporary variable. To swap numbers without using extra variable see another code below.

Swapping using temporary or third variable

```
import java.util.Scanner;

class SwapNumbers {
public static void main(String args[]) {

System .out.println("Enter x
and y");
Scanner in = new Scanner(System.in);

x = in.nextInt();
y = in.nextInt();
int x, y, temp;
System.out.println("Before
Swapping\nx = "+x+"\ny = "+y);

temp = x;
x = y;
y = temp;

System.out.println("After Swapping\nx = "+x+"\ny = "+y);
}
}
```

Output of program:

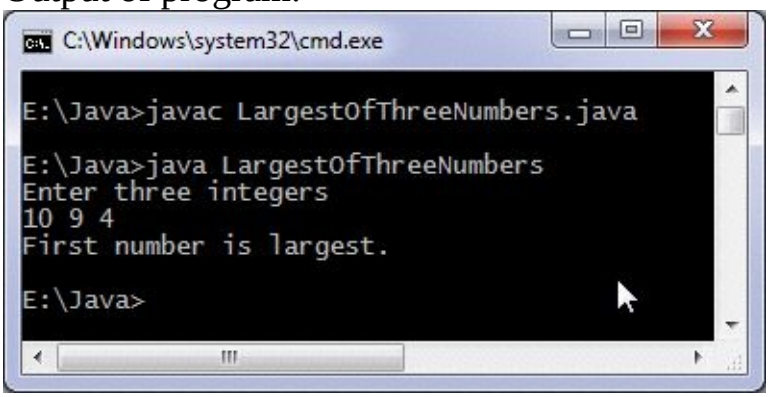
Java program to find largest of three numbers

This java program finds largest of three numbers and then prints it. If the entered numbers are unequal then “numbers are not distinct” is printed.

Java programming source code

```
import java.util.Scanner;
class LargestOfThreeNumbers {
public static void main(String args[]) {

System .out.println(“Enter
three integers “);
Scanner in = new Scanner(System.in);

x = in.nextInt();
y = in.nextInt();
z = in.nextInt();

if ( x > y && x > z )

System .out.println(“First
number is largest.”);
else if ( y > x && y > z)

int x, y, z;

System .out.println(“Second
number is largest.”);
else if ( z > x && z > y)

System .out.println(“Third
number is largest.”);
else

System .out.println(“Entered
numbers are not distinct.”);
}

}
```

Output of program:



If you want to find out largest of a list of numbers say 10 integers then using above approach is not easy, instead you can use array data structure.

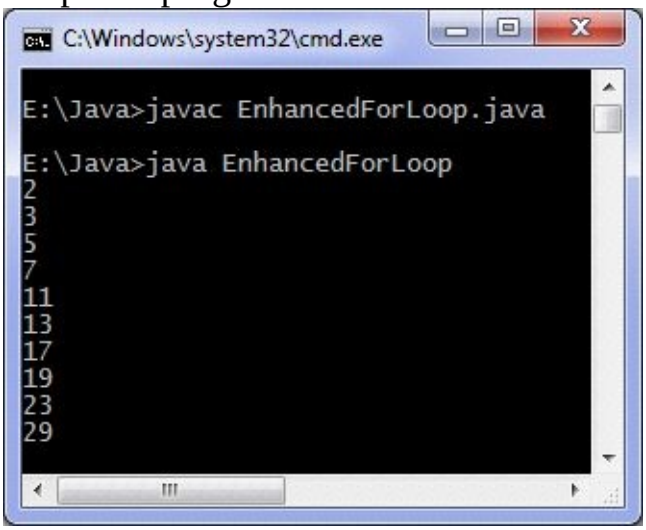
Enhanced for loop java

Enhanced for loop java: Enhanced for
loop is useful when scanning the array instead of using for loop. Syntax

of enhanced for loop is:
for (data_type variable:
array_name)
Here array_name is the name of array.
Java enhanced for loop integer
array

```
class EnhancedForLoop { public static void
main(String[] args) {
int primes[] = { 2, 3,
5 , 7, 11, 13, 17, 19, 23,
29};
for (int t: primes)
{ System.out.println(t);
} }
}
```

Output of program:



Java exception handling tutorial with example programs

Java exception handling tutorial: In this tutorial we will learn how to handle exception with the help of suitable examples. Exceptions are errors which occur when the program is executing. Consider the Java program below which divides two integers.

```
import java.util.Scanner; class Division {
public static void main(String[] args) { int a, b, result;
Scanner input = new Scanner(System.in);

System .out.println(“Input
two integers”);
a = input.nextInt();
b = input.nextInt();
result = a / b;

System .out.println(“Result
= ” + result);
}

}
```

Now we compile and execute the above code two times, see the output of program in two cases:

In the second case we are dividing a by zero which is not allowed in mathematics, so a run time error will occur i.e. an exception will occur. If

we write programs in this way then they will be terminated abnormally and user who is executing our program or application will not be happy. This occurs because input of user is not valid so we have to take a preventive action and the best thing will be to notify the user that it is not allowed or any other meaningful message which is relevant according to context. You can see the information displayed when exception occurs it includes name of thread, file name, line of code (14 in this case) at which exception occurred, name of exception (ArithmeticException) and it’s description(‘/ by zero’). Note that exceptions don’t occur only because of invalid input only there are other reasons which are beyond of programmer control such as stack overflow exception, out of memory exception when an application requires memory larger than what is available.

Java provides a powerful way to handle such exceptions which is known as exception handling. In it we write vulnerable code i.e. code which can throw exception in a separate

block called as
try
block and
exception handling code in another
block called
catch

block. Following
modified code handles the exception. Java exception handling example

```
class Division { public static void main(String[] args) {
int a, b, result;
Scanner input = new Scanner(System.in);

System .out.println(“Input
two integers”);
a = input.nextInt();
b = input.nextInt();
// try block

try {
result = a / b;

System .out.println(“Result =
” + result);
}

// catch block
catch (ArithmeticException e) {

System .out.println(“Exceptio n caught: Division by
zero.”);

} }
}
```

Whenever an exception is caught corresponding catch block is executed, For example above code catches ArithmeticException only. If some other kind of exception is thrown it will not be caught so it’s the programmer work to take care of all exceptions as in our try block we are performing arithmetic so we are capturing only arithmetic exceptions. A simple way to capture any exception is to use an object of Exception class as other classes inherit Exception class, see another example below:

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

String languages[] =
{ “C”, “C++”, “Java”,
“Perl”, “Python” };
try {
for (int c = 1; c <= 5;

c++) {
```

```
System .out.println(languages
[c]);
} }
catch (Exception e)
{ System.out.println(e);
} }
}
```

Output of program:

```
C ++
Java
Perl
Python
java.lang.ArrayIndexOutOfBou
ndsException: 5
```

Here our catch block capture an exception which occurs because we are trying to access an array element which does not exists (languages[5] in this case). Once an exception is thrown control comes out of try block and remaining instructions of try block will not be executed. At compilation time syntax and semantics checking is done and code

is not executed on machine so
exceptions can only be detected at run time.
Finally block in Java

Finally block is always executed whether an exception is thrown or not.

```
class Allocate { public static void
main (String[] args) {
try {
long data[] = new
long[1000000000]; }
catch (Exception e)
{ System.out.println(e);
}
finally {
System .out.println("finally
block will execute
always.");
} }
}
```

Output of program:

```
finally block will execute always.  Exception in thread
```

```
“main”
java. lang.OutOfMemoryError:
Java heap space
at
Allocate.main(Allocate.java:
5)
```

Exception occurred because we try to allocate a large amount of memory which is not available. This amount of memory may be available on your system if this is the case try increasing the amount of memory to allocate through the program.

14.Java Programs part 4

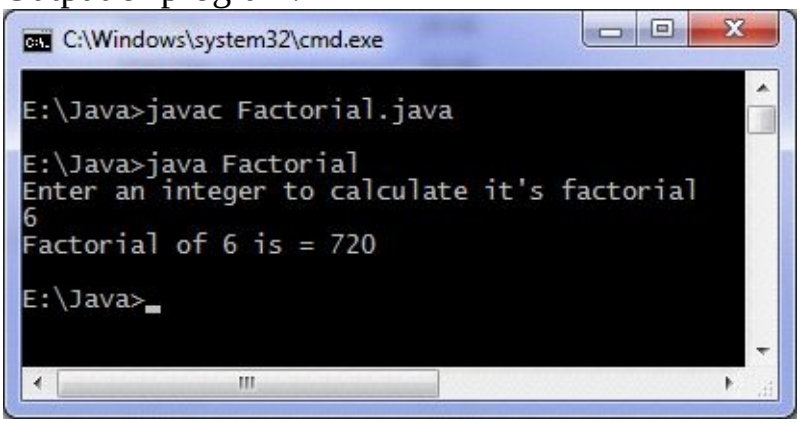
Java program to find factorial

This java program finds factorial of a number. Entered number is checked first if its negative then an error message is printed.

Java programming code

```
import java.util.Scanner;
class Factorial {
public static void main(String args[]) {

System .out.println(“Enter an
integer to calculate it’s
factorial”);
Scanner in = new Scanner(System.in);

n = in.nextInt();
if ( n < 0 )

System .out.println(“Number
should be non-negative.”);
else

{
int n, c, fact = 1;
for ( c = 1 ; c <= n ; c++ )

fact = fact*c;
System.out.println(“Factoria
l of “+n+” is = “+fact);
} }

}
```

Output of program:



You can also find factorial using recursion, in the code fact is an integer variable so only factorial of small numbers will be correctly displayed or which fits in 4 bytes. For large numbers you can use long data type.

Java program for calculating factorial of large numbers

Above program does not give correct

result for calculating factorial of say 20. Because 20! is a large number and cant be stored in integer data type which is of 4 bytes. To calculate factorial of say hundred we use BigInteger class of java.math package.

```
import java.util.Scanner; import java.math.BigInteger;

class BigFactorial {
public static void main(String args[]) {
BigInteger inc = new BigInteger("1"); BigInteger fact = new
BigInteger("1"); Scanner input = new Scanner(System.in);

System .out.println("Input an
integer");
n = input.nextInt();
for (c = 1; c <= n; c++)

{
fact = fact.multiply(inc);

int n, c;
inc =

inc. add(BigInteger.ONE);
}
System.out.println(n +

"! = " + fact);
}
}
```

We run the above java program to calculate 100 factorial and following output is obtained.

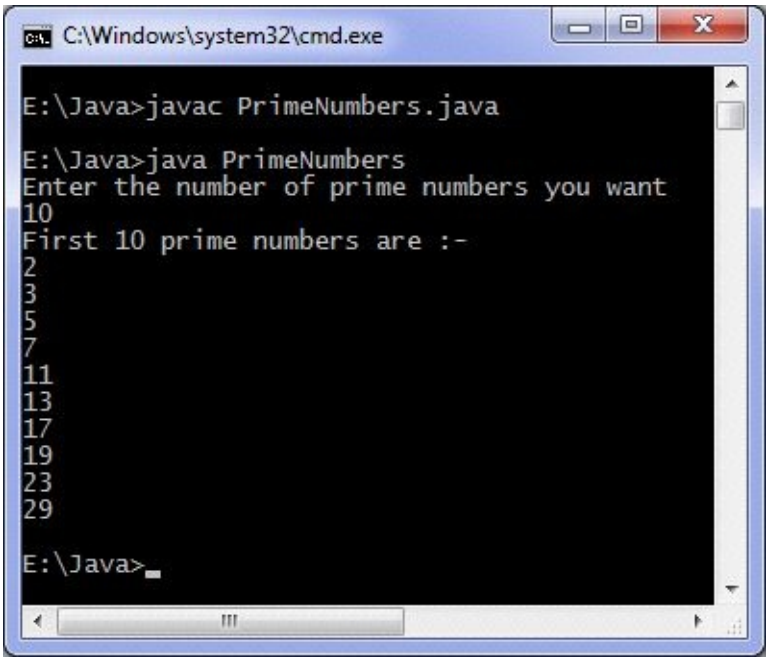
```
Input an integer
100
100! =
933262154439441526816992388562667004907159682643816214685929638952175999932299156089414639761565182862536979208272237582511852109168640000000000000000
```

Java program print prime numbers

This java program prints prime numbers, number of prime numbers required is asked from the user. Remember that smallest prime
number is 2.
Java programming code

```
import java.util.*;
class PrimeNumbers {
public static void main(String args[]) {
int n, status = 1, num = 3;
Scanner in = new Scanner(System.in);

System .out.println("Enter
the number of prime numbers
you want");
n = in.nextInt(); if (n >= 1)

{

System .out.println("First
"+n+" prime numbers
are :-");
System.out.println(2);

}
for ( int count = 2 ; count <=n ; ) {
for ( int j = 2 ; j <= Math.sqrt(num) ; j ++ )
0)
{
if ( num%j ==

{
break; }

}
if ( status != 0 ) {

System .out.println(num);
count++;
}

status = 1;
num++; }
}

status = 0;
}
```

Output of program:



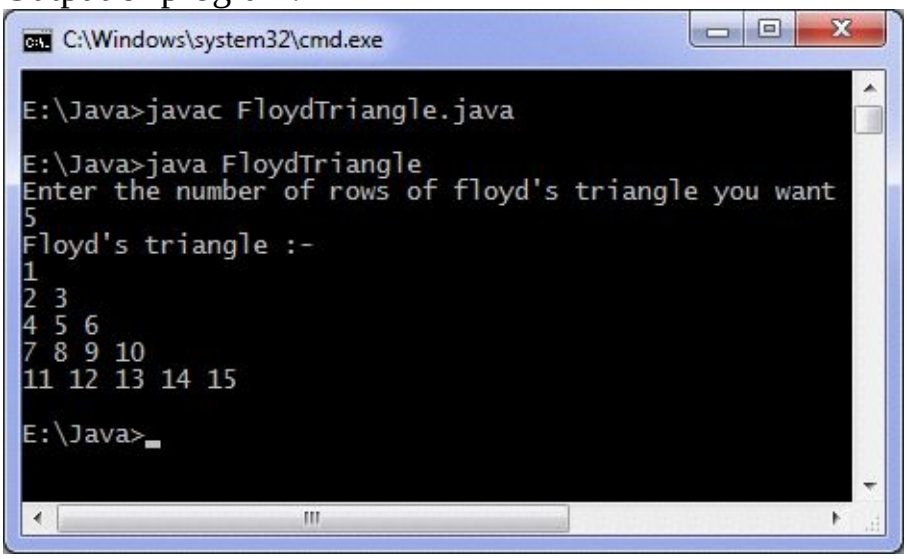
We have used sqrt method of Math package which find square root of a number. To check if an integer(say n) is prime you can check if it is divisible by any integer from 2 to (n-1) or check from 2 to sqrt(n), first one is less efficient and will take more time.

Java program to print Floyd’s triangle

This java program prints Floyd’s triangle.
Java programming source code

```
import java.util.Scanner;
class FloydTriangle {
public static void main(String args[]) {
Scanner in = new Scanner(System.in);

System .out.println(“Enter
the number of rows of
floyd’s triangle you want”);

n = in.nextInt();
System.out.println(“Floyd’s
triangle :-”);
for ( c = 1 ; c <= n ;

c++ )
{
for ( d = 1 ; d <= c ; d++ )
{

System .out.print(num+” “);
num++;
}

System .out.println();
}
}

int n, num = 1, c, d;


 }
```

Output of program:



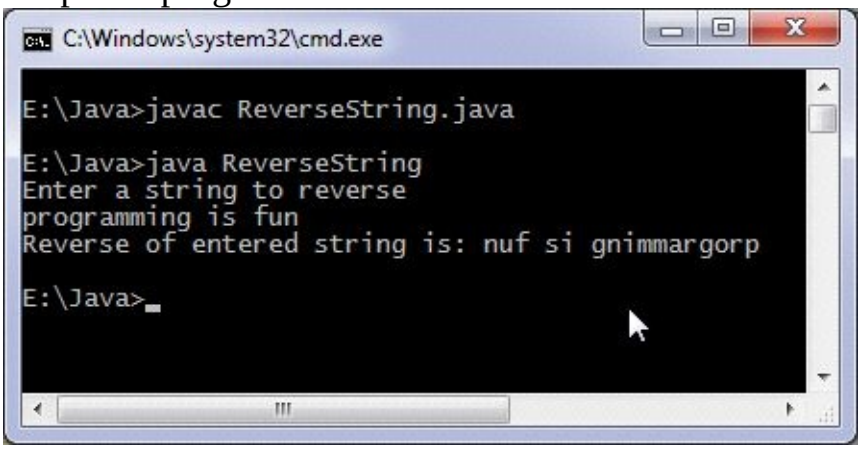
In Floyd triangle there are n integers in the nth row and a total of (n(n+1))/ 2 integers in n rows. This is a simple pattern to print but helpful in learning how to create other patterns. Key to develop pattern is using nested loops appropriately.

Java program to reverse a string

This java program reverses a string entered by the user. We use charAt method to extract characters from the string and append them in reverse
order to reverse the entered string. Java programming code

```
import java.util.*;
class ReverseString {
public static void main(String args[]) {

String original,
reverse = "";
Scanner in = new Scanner(System.in);

System.out.println("Enter a
string to reverse");
original =
in.nextLine();

reverse = reverse +
original.charAt(i);
System.out.println("Reverse
of entered string is:
"+reverse);
}

 }
```

Output of program:



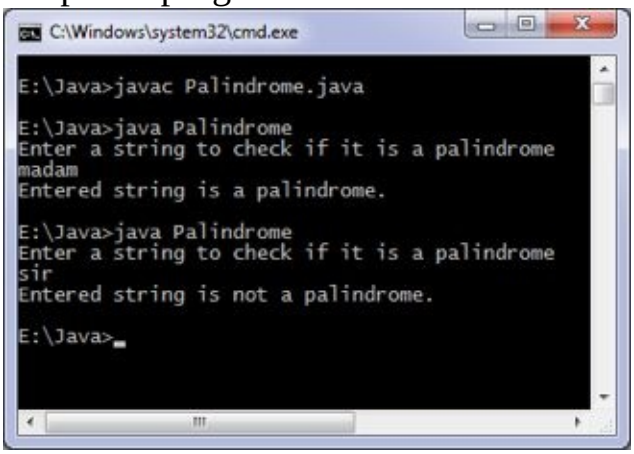
Java program to check palindrome

Java palindrome program: Java program to check if a string is a palindrome or not. Remember a string is a palindrome if it remains unchanged when reversed, for example “dad” is a palindrome as reverse of “dad” is “dad” whereas “program” is not a palindrome. Some other palindrome strings are “mom”, “madam”, “abcba”.

Java programming source code

```
import java.util.*;
class Palindrome {
public static void main(String args[]) {

String original,
reverse = "";
Scanner in = new Scanner(System.in);

System .out.println(“Enter a
string to check if it is a
palindrome”);

original =
in.nextLine();

reverse = reverse + original.charAt(i);
if

( original.equals(reverse)) System.out.println(“Entered
string is a palindrome.”); else

System .out.println(“Entered string is not a
palindrome.”);

}
}
```

Output of program:



Interface in Java

Interface in Java: Java interfaces are like Java classes but they contain only static final constants and declaration of methods. Methods are not defined and classes which implements an interface must define the body of method(s) of interface(s). Final constants can't be modified once they are initialized; final, interface, extend and implements are Java keywords.

Declaration of interface:

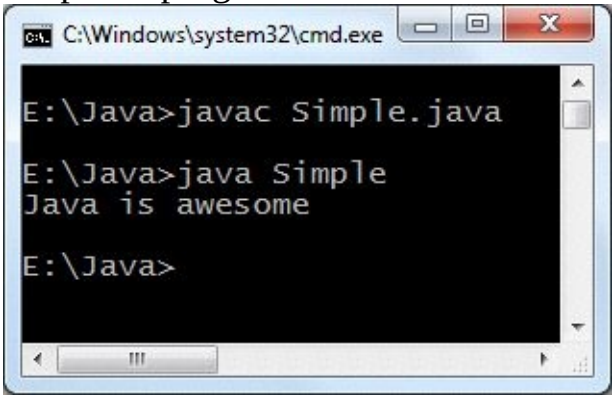
```
interface InterfaceName { // constants declaration //
methods declaration

}
```

Interface program in Java

In our program we create an interface named Info which contains a constant and a method declaration. We create a class which implements this interface by defining the method declared inside it.

```
interface Info { static final String language = "Java";
public void display(); }
class Simple implements Info {
public static void main(String []args) { Simple obj = new
Simple();

obj. display();
}
// Defining method

declared in interface
public void display() {

System .out.println(language
+ " is awesome");
}

}
```

Output of program:



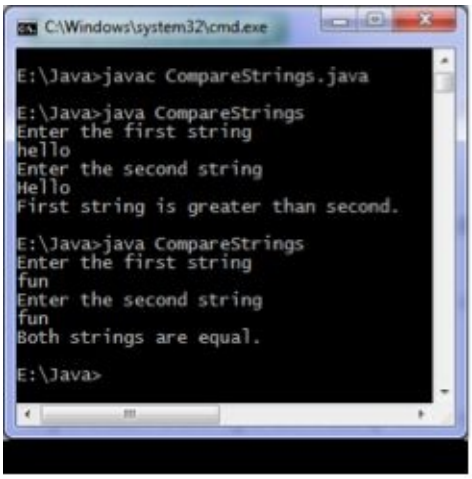
Java program to compare two strings

This program compare strings i.e test whether two strings are equal or not, compareTo method of String class is used to test equality of two String class objects. compareTo method is case sensitive i.e “java” and “Java” are two different strings if you use compareTo method. If you wish to

compare strings but ignoring the case
then use compareToIgnoreCase method.
Java programming code

```
import java.util.Scanner;
class CompareStrings {
public static void main(String args[]) {
Scanner in = new Scanner(System.in);

System.out.println(“Enter
the first string”);
s1 = in.nextLine();
System.out.println(“Enter
the second string”);

s2 = in.nextLine(); if ( s1.compareTo(s2) >0)

System .out.println(“First
string is greater than
second.”);
else if

( s1.compareTo(s2) < 0 )
String s1, s2;
System.out.println(“First
string is smaller than
second.”);
else

System .out.println(“Both
strings are equal.”);
}

}
```

Output of program:



String ‘hello’ is greater than ‘Hello’ as ASCII value of ‘h’ is greater than ‘H’. To check two strings for equality you can use equals method which returns true if strings are equal otherwise false.

15. Java Programs part 5

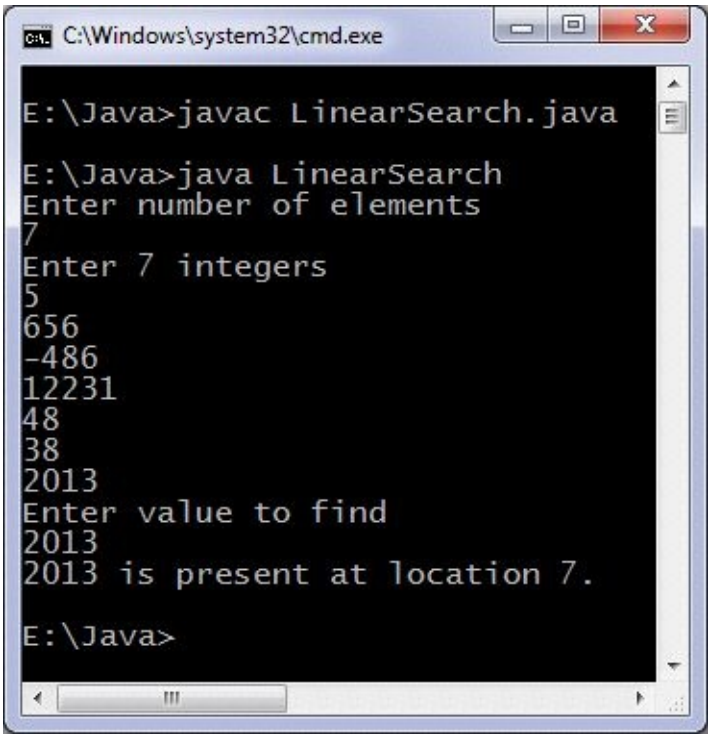
Java program for linear search

Java program for linear search: Linear search is very simple, To check if an element is present in the given list we compare search element with every element in the list. If the number is found then success occurs otherwise the list doesn't contain the element we are searching.

Java programming code

```
import java.util.Scanner;
class LinearSearch {
public static void main(String args[]) {
int c, n, search, array[];
Scanner in = new Scanner(System.in);

System.out.println("Enter
number of elements");
n = in.nextInt();
array = new int[n];

System .out.println("Enter "
+ n + " integers");
for (c = 0; c < n; c++) array[c] =

in. nextInt();
System.out.println("Enter
value to find");

search = in.nextInt();
for (c = 0; c < n; c++) {
if (array[c] == search) /* Searching element is present */
{

System .out.println(search +
" is present at location " +
(c + 1) + ".");
break; }

}
if (c == n) /* Searching element is absent */

System .out.println(search +
" is not present in
array.");
}

}
```

Output of program:



Above code locate first instance of element to found, you can modify it for multiple occurrence of same element and count how many times it occur in the list. Similarly you can find if an alphabet is present in a string.

Java program for binary search

Java program for binary search: This code implements binary search algorithm. Please note input numbers

must be in *ascending* order.Java programming code

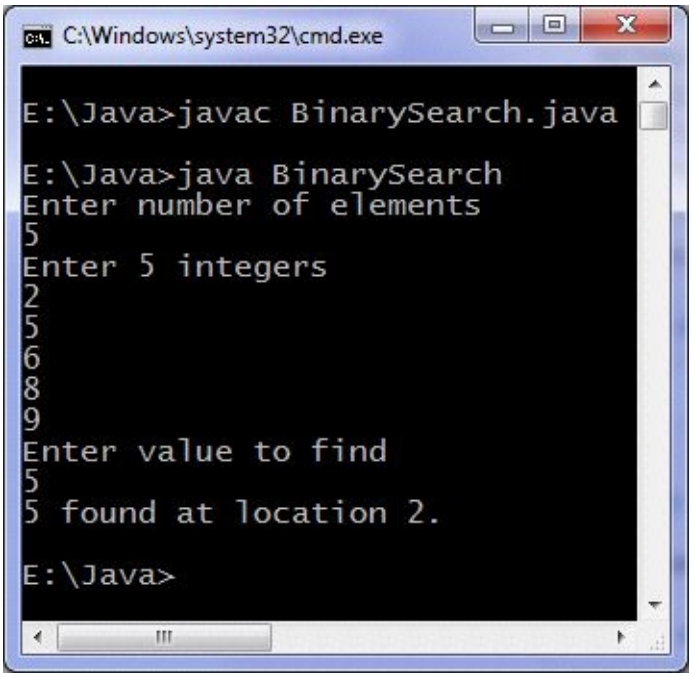
```
import java.util.Scanner;
class BinarySearch {
public static void main(String args[]) {
int c, first, last, middle, n, search, array[];
Scanner in = new Scanner(System.in);
System.out.println("Enter
number of elements");
n = in.nextInt();
array = new int[n];
System.out.println("Enter "
+ n + " integers");
for (c = 0; c < n; c++) array[c] =
in. nextInt();
System.out.println("Enter
value to find");
search = in.nextInt();
first = 0;
last =n-1;
middle = (first + last)/
2 ;
if ( array[middle] < search ) first = middle + 1; else if
( array[middle] == search ) {
System.out.println(search +
" found at location " +
(middle + 1) + ".");
break; }
else
last = middle - 1;
middle = (first +
last)/2;
}
if ( first > last )
```

```
System.out.println(search + " is not present in the
list.\n");
while( first <= last ) {
```

[redacted]

```
}
}
```

Output of program:

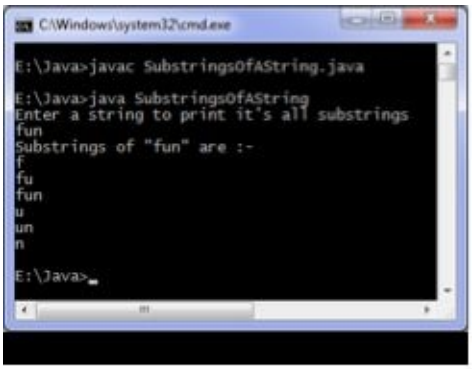


Java program to find all substrings of a string

Java program to find substrings of a string :- This program find all substrings of a string and the prints them. For example substrings of “fun” are :- “f”, “fu”, “fun”, “u”, “un” and “n”. substring method of String class is used to find substring. Java code to

print substrings of a string is given
below.
Java programing code

```
import java.util.Scanner;
class SubstringsOfAString {
public static void main(String args[]) {
Scanner in = new Scanner(System.in);

System .out.println(“Enter a
string to print it’s all
substrings”);

string =
in.nextLine();
length =
string.length();
System.out.println(“Substrin gs of ””+string+””
are :-”);

for( c = 0 ; c < length ; c++ )

{
String string, sub; int i, c, length; for( i = 1 ; i <=

length - c ; i++ )

{
sub =
string.substring(c, c+i);
System.out.println(sub);

}
} }
}
```



For a string of length n there will be (n(n+1))/2 non empty substrings and one more which is empty string. Empty string is considered to be substring of every string also known as NULL string.

Java program to generate random numbers

Java program to generate random numbers: This code generates random numbers in range 0 to 100 (both inclusive).

Java programming code

```
import java.util.*;
class RandomNumbers { public static void main(String[] args)
{ int c;

Random t = new Random(); // random integers in

[0, 100]
for (c = 1; c <= 10; c+ +) {

System .out.println(t.nextInt
(100));
} }

}
```

Download Random Numbers program class file.

nextInt(c) method returns next integer in 0 to c (both inclusive), c must be positive. To generate random float's use nextFloat which returns float between 0.0 to 1.0.

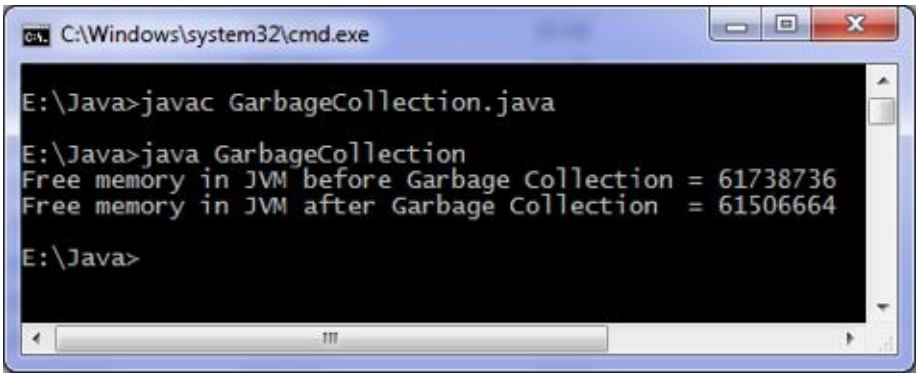
Java program to perform garbage collection

This program performs garbage collection. Free memory in java virtual machine is printed and then garbage collection is done using gc method of RunTime class,

freeMemory method returns amount
of free memory in jvm, getRunTime method is used to get reference of current RunTime object.

Java programming source code

```
import java.util.*;
class GarbageCollection {

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

Runtime rs =
Runtime.getRuntime();
System.out.println("Free memory in JVM before Garbage
Collection =
"+rs.freeMemory());
rs.gc();

System .out.println("Free memory in JVM after Garbage
Collection =
"+rs.freeMemory());
}

}
```



Obviously the amount of available after garbage collection will be different on your computer. Numbers are not important, what is important is that amount of memory available is more than before. You can use this code in your program or projects which uses large amount of memory or where frequently new objects are created but are required for a short span of time.

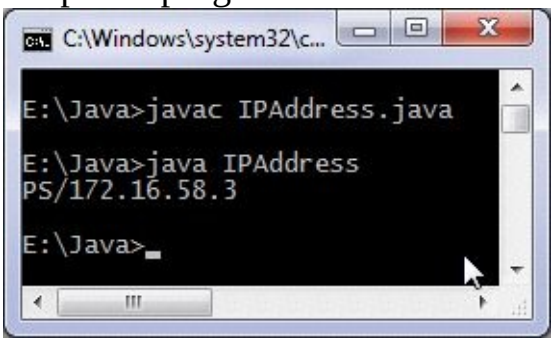
Java program to get ip address

This program prints IP or internet protocol address of your computer system. InetAddress class of java.net package is used, getLocalHost method returns InetAddress object which represents local host.

Java programming source code

```
import java.net.InetAddress;
class IPAddress {

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

System .out.println(InetAddre
ss.getLocalHost());
}

}
```

Output of program:



Output of code prints computer name/ IP address of computer. Java has a very vast Networking API and can be used to develop network applications.

16. Java Programs part 6

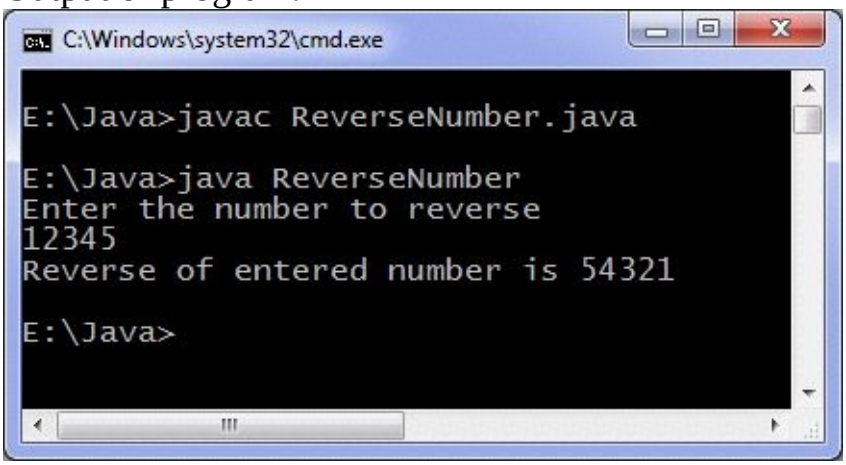
Java program to reverse number

This program prints reverse of a number i.e. if the input is 951 then output will be 159.

Java programming source code

```
import java.util.Scanner;
class ReverseNumber {
public static void main(String args[]) {

System .out.println(“Enter
the number to reverse”);
Scanner in = new Scanner(System.in);

n = in.nextInt();
while( n != 0 ) {
* 10;
+ n%10;

reverse = reverse
reverse = reverse
int n, reverse = 0;

n = n/10; }

System .out.println(“Reverse
of entered number is
“+reverse);
}

}
```

Output of program:



You can also reverse or invert a number using recursion. You can use this code to check if a number is palindrome or not, if the reverse of an integer is equal to integer then it’s a palindrome number else not.

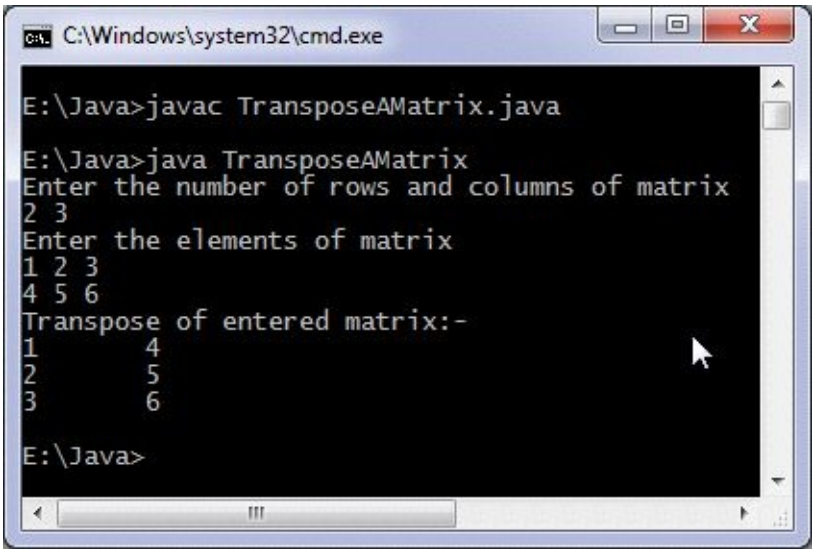
Java program to transpose matrix

This java program find transpose of a matrix of any order.

Java programming source code

```
import java.util.Scanner;
class TransposeAMatrix {
public static void main(String args[]) {

Scanner (System.in);
System.out.println(“Enter
the number of rows and
columns of matrix”);

m = in.nextInt();
n = in.nextInt();
int matrix[][] = new int[m][n];

System .out.println(“Enter
the elements of matrix”);
for ( c = 0 ; c < m ; c++ )

for ( d = 0 ; d < n ; d++ )

matrix [c][d] =
in.nextInt();
int transpose[][] = new int[n][m];

int m, n, c, d; Scanner in = new for ( c = 0 ; c < m ; c++ )

{
for ( d = 0 ; d <

n ; d++ )
= matrix[c][d];
transpose[d][c]

}
System.out.println(“Transpos
e of entered matrix:-”); for ( c = 0 ; c < n ;

c++ )
{

for ( d = 0 ; d < m ; d++ )
System.out.print(transpose[c ][d]+”\t”);
System.out.print(”\n”); }
}
}
```

Output of program:



This code can be used to check if a matrix symmetric or not, just compare the matrix with it's transpose if they are same then it's symmetric otherwise non symmetric, also it's useful for calculating orthogonality of a matrix.

Java program to multiply two matrices

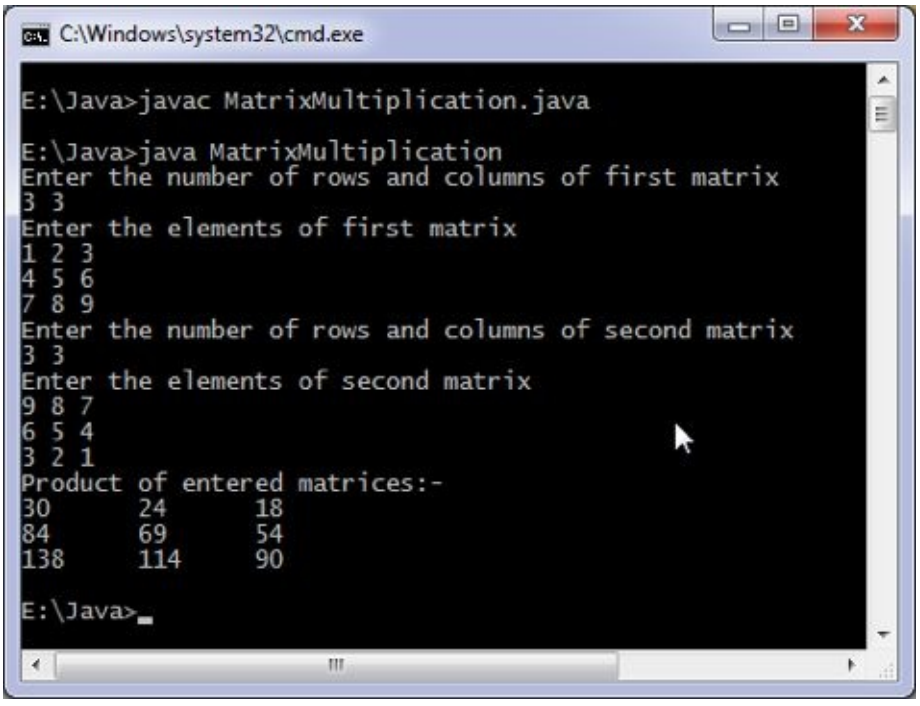
This java program multiply two matrices. Before multiplication matrices are checked whether they can be multiplied or not.

Java programming code

```
import java.util.Scanner; class
MatrixMultiplication

{
public static void main(String args[]) {
int m, n, p, q, sum = 0, c, d, k; Scanner in = new
Scanner(System.in);

System .out.println("Enter
the number of rows and
columns of first matrix");

m = in.nextInt();
n = in.nextInt();
int first[][] = new int[m][n];

System .out.println("Enter
the elements of first
matrix");
for ( c = 0 ; c < m ; c++ )

for ( d = 0 ; d < n ; d++ )

first [c][d] =
in.nextInt();
System.out.println("Enter
the number of rows and
columns of second matrix");

p = in.nextInt();
q = in.nextInt();
if ( n != p )

System .out.println("Matrices with entered orders can't be
multiplied with each
other.");
else

{
int second[][] =
new int[p][q];
int multiply[][] =
```

```
new int[m][q];
System .out.println("Enter the elements of second
matrix");
for ( c = 0 ; c < p ; c++ )
< q ; d++ )
= in.nextInt();
for ( d = 0 ; d second[c][d]
for ( c = 0 ; c < m ; c++ ) {
for ( d = 0 ; d
< q ; d++ )
{
for ( k = 0 ; k < p ; k++ ) {
sum = sum + first[c][k]*second[k][d]; [d] = sum; }
}
multiply[c]
sum = 0; }
System .out.println("Product of entered matrices:-"); for (
c = 0 ; c < m ; c++ )
{
for ( d = 0 ; d
< q ; d++ )
System.out.print(multiply[c] [d]+"\t");
System .out.print("\n"); } } }
}
```

Output of program:



This is a basic method of multiplication, there are more efficient algorithms available. Also this approach is not recommended for sparse matrices which contains a large number of elements as zero.

Java program to open Notepad

How to open Notepad through java program: Notepad is a text editor which comes with Windows operating system, It is used for creating and editing text files. You may be developing java programs in it but you can also open it using your java code.

How to open notepad using Java program

```
import java.util.*; import java.io.*; class Notepad { public static void

main (String[] args) {
Runtime rs =
Runtime.getRuntime();
try { rs.exec(“notepad”);

}
catch (IOException e) {
System.out.println(e);
}

}
}
```

Download Notepad program.

Explanation of code: getRunTime method is used to get reference of current RunTime object, exec method can be used to execute commands. You can also specify a file while opening notepad such as exec(“notepad programming.txt”) where ‘programming.txt’ is the file you wish to open, if the file doesn’t exist in current working directory then a dialog box will be displayed to

create file. You can launch other

applications using exec method, for example exec(“calc”) will launch calculator application. If an application is present in a directory which is not set in environment variable PATH then you can specify complete path of application. If you are still using Notepad for Java development it is recommended to switch to some advanced text editor like Notepad++ or use a development IDE.

How to find a digit string from the given alphanumeric string.

```
import java.util.regex.Matcher;
import java.util.regex.Pattern;
public class RegexMatches
{

main ( String args[] ){
// String to be scanned to
find the pattern.
String line = “This order
was placed for QT3000! OK?”;
String pattern = “(.*)(\d+)
(.*)”;
// Create a Pattern object Pattern r =
Pattern.compile(pattern);
// Now create matcher
object.
Matcher m = r.matcher(line); if (m.find( )) {
System.out.println(“Found
value: ” + m.group(0) );
System.out.println(“Found
value: ” + m.group(1) );
System.out.println(“Found
value: ” + m.group(2) );
} else {
System.out.println(“NO
MATCH”); }

} }
public static void
```

This would produce the following result:

```
Found value: This order was placed for QT3000! OK?
Found value: This order was placed for QT300
Found value: 0
```

How to search a word inside a string ?

Solution:

This example shows how we can search a word within a String object using indexOf() method which returns a position index of a word within the string if found. Otherwise it returns -1.

Solution:

```
public class SearchStringEmp{
public static void
main(String[] args) {
String strOrig = “Hello
readers”;
int intIndex =
strOrig.indexOf(“Hello”);
if(intIndex == - 1){
System.out.println(“Hello
not found”);
}else{
System.out.println(“Found
Hello at index ”
+ intIndex);
}
} }
```

Result:

The above code sample will
Result:
produce the following result.

How to optimize string concatenation ?

Solution:

Following example shows performance of concatenation by using “+” operator and StringBuffer.append() method.

Found Hello at index 0

```
public class StringConcatenate{ public static void
main(String[] args){
long startTime =
System.currentTimeMillis(); for(int i=0;i<5000;i++){ String result = “This is” + “testing the”
+ “difference”+ “between” + “String”+ “and”+
“StringBuffer”;
}
long endTime =
System.currentTimeMillis(); System.out.println(“Time
taken for string”
+ “concatenation using +
operator : ”
+ (endTime - startTime)+ ”
ms”);

long startTime1 =
System.currentTimeMillis();
for(int i=0;i<5000;i++){

StringBuffer result = new
StringBuffer();
result.append(“This is”);
result.append(“testing
the”);

result .append(“difference”);
result.append(“between”);
result.append(“String”);
result.append(“and”);

result .append(“StringBuffer”);
}
long endTime1 =

System .currentTimeMillis();
System.out.println(“Time
taken for String concatenation”
+ “using StringBuffer : ”
+ (endTime1 - startTime1)+ ”
ms”);
} }
```

Result:

The above code sample will produce the following result.The result may vary.

Result:

```
Time taken for stringconcatenation
using + operator : 0 ms

Time taken for String
```

concatenationusing StringBuffer :
22 ms

How to merge two arrays ?

Solution:

This example shows how to merge two arrays into a single array by the use of list.Addall(array1.asList(array2) method of List class and Arrays.toString () method of Array class.

Solution:

```
import java.util.ArrayList;
import java.util.Arrays;
import java.util.List;
public class Main {

public static void main(String
args[]) {

String a[] = { “A”, “E”,
“I” };
String b[] = { “O”, “U” };
List list = new
ArrayList(Arrays.asList(a));
list.addAll(Arrays.asList(b));
Object[] c = list.toArray();
System.out.println(Arrays.toString
(c));
} }
```

Result:

The above code sample will produce the following result.

How to check if two arrays are equal or not?

Solution:

Following example shows how to use equals () method of Arrays to check if two arrays are equal or not.

Result:
[A, E, I, O, U]
Solution:

```
import java.util.Arrays;
public class Main {
public static void
main(String[] args) throws

Exception {
int[] ary = {1,2,3,4,5,6};
int[] ary1 = {1,2,3,4,5,6};
int[] ary2 = {1,2,3,4};

System .out.println("Is array
1 equal to array 2?? "
+Arrays.equals(ary, ary1));
System.out.println("Is array
1 equal to array 3?? "

+Arrays.equals(ary, ary2));
}
}
```

Result:

The above code sample will produce the following result.

How to use method overriding in Inheritance for subclasses ?

Solution:

This example demonstrates the way of method overriding by subclasses with different number and type of parameters.

Result :

Is array 1 equal to array 2?? true
Is array 1 equal to array 3??
false

Solution :

```
public class Findareas{
public static void main (String
[]agrs){
Figure f= new Figure(10 ,
10);

Rectangle r= new
Rectangle(9 , 5);
Figure figref;
figref=f;
System.out.println("Area is :"+figref.area());
figref=r;
System.out.println("Area is :"+figref.area());
} }

class Figure{
double dim1;
double dim2;
Figure(double a , double b) {

dim1=a;
dim2=b; }
Double area() {

System .out.println("Inside area for figure.");
return(dim1*dim2);
} }

class Rectangle extends Figure {

Rectangle (double a, double b) {
super(a ,b);
}
Double area() {
System.out.println("Inside
area for rectangle.");
return(dim1*dim2);
} }
```

Result:

The above code sample will produce the following result.

Result:

Inside area for figure.
Area is :100.0
Inside area for rectangle. Area is :45.0

How to use for and foreach loops to display elements of an array.

Solution:

This example displays an integer array using for loop & foreach loops.

```
public class Main {
public static void
main(String[] args) {
int[] intary = { 1,2,3,4}; forDisplay(intary);
foreachDisplay(intary);
}

public static void
forDisplay(int[] a){

System .out.println(“Display an array using for loop”);
for (int i = 0; i <
a.length; i++) {
System.out.print(a[i] + ” “);
}

System.out.println(); }

public static void
foreachDisplay(int[] data){

System .out.println(“Display an array using for
each loop”); for(inta :data){

System.out.print(a+ ” “); }
} }
```

Result:

The above code sample will produce the following result.

```
Display an array using for loop 1234
Display an array using for each loop
1234
```

How to implement stack?

Solution:

Following example shows how to implement stack by creating user defined push() method for entering elements and pop() method for retriving elements

[redacted]from the stack.

```
public class MyStack {
private int maxSize;
private long[] stackArray;
private int top;
public MyStack(int s) {

maxSize = s;
stackArray = new
long[maxSize];
top = 1; }

public void push(long j) { stackArray[++top] = j;
}
public long pop() {
return stackArray[top—];
}
public long peek() {
return stackArray[top];
}
public boolean isEmpty() { return (top == 1);
}
public boolean isFull() { return (top == maxSize - 1); }

public static void
main(String[] args) {

MyStack theStack = new MyStack(10);
theStack.push(10);
theStack.push(20);
theStack.push(30);
theStack.push(40);
theStack.push(50);
while (!theStack.isEmpty()) {

long value =

theStack .pop();
System.out.print(value);
System.out.print(" ");

}
System.out.println("");
}
}
```

Result:

The above code sample will produce the following result.

How to implement Queue ?

Solution:

Following example shows how to implement a queue in an employee structure.

[redacted]50 40 30 20 10

[redacted]

```
import java.util.LinkedList;
class GenQueue<E> {
private LinkedList<E> list =
new LinkedList<E>();
public void enqueue(E item) {
list.addLast(item);
}
public E dequeue() {
return list.poll();
}
public boolean hasItems() { return !list.isEmpty();
}
public int size() {
return list.size();
}
public void addItems(GenQueue<? extends E> q) {
while (q.hasItems())
list.addLast(q.dequeue());
}
}
public class GenQueueTest {
public static void
main(String[] args) {
GenQueue<Employee> empList; empList = new
GenQueue<Employee>();
GenQueue<HourlyEmployee>
hList;
hList = new
GenQueue<HourlyEmployee>(); hList.enqueue(new
HourlyEmployee(“T”, “D”));
hList.enqueue(new
HourlyEmployee(“G”, “B”));
hList.enqueue(new
HourlyEmployee(“F”, “S”));
empList.addItems(hList); System.out.println(“The employees’ names are:”);
while (empList.hasItems()) { Employee emp =
empList.dequeue();
System.out.println(emp.firstName + ””
} }
+ emp.lastName);
}
class Employee {
public String lastName;
public String firstName;
public Employee() {
}
public Employee(String last,
```

```
String first) {
this.lastName = last;
this.firstName = first;
}
public String toString() { return firstName + ” ” +
lastName ; }
}
class HourlyEmployee extends
Employee {
public double hourlyRate; public HourlyEmployee(String
last, String first) {
super(last, first);
} }
```

Result:

The above code sample will produce the following result.

Result :

```
The employees’ name are: TD
GB
FS
```

17. Java Interview Questions part 1

1) What is difference between JDK,JRE and JVM?

JVM

JVM is an acronym for Java Virtual Machine, it is an abstract machine which provides the runtime environment in which java bytecode can be executed. It is a specification.

JVMs are available for many hardware and software platforms (so JVM is platform dependent).

JRE

JRE stands for Java Runtime Environment. It is the implementation of JVM.

JDK

JDK is an acronym for Java
Development Kit. It physically exists.
It contains JRE + development tools.

2) How many types of memory areas are allocated by JVM?

Many types:
1. Class(Method) Area
2. Heap

3. Stack
4. Program Counter Register
5. Native Method Stack

3) What is JIT compiler?

Just-In-Time(JIT) compiler: It is used to improve the performance. JIT compiles parts of the byte code that have similar functionality at the same time, and hence reduces the amount of time needed for compilation.Here the term “compiler” refers to a translator from the instruction set of a Java virtual machine (JVM) to the instruction set of a specific CPU.

4) What is platform?

A platform is basically the hardware or software environment in which a program runs. There are two types of platforms software-based and hardwarebased. Java provides software-based platform.

5) What is the main difference between Java platform and other platforms?

The Java platform differs from most other platforms in the sense that it’s a software-based platform that runs on top of other hardware-based platforms.It has two components:

1. Runtime Environment
2. API(Application Programming
Interface)

6) What gives Java its ‘write once and run anywhere’ nature?

The bytecode. Java is compiled to be a byte code which is the intermediate language between source code and machine code. This byte code is not platform specific and hence can be fed to any platform.

7) What is classloader?

The classloader is a subsystem of JVM that is used to load classes and interfaces.There are many types of classloaders e.g. Bootstrap classloader, Extension classloader, System classloader, Plugin classloader etc.

8) Is Empty .java file name a valid source file name?

Yes, save your java file by .java only, compile it by **javac .java** and run by **java**

yourclassnameLet’s take a simple example:

```
1. //save by .java only
2. class A{
3.
public static void main(String
args[]){
4.
System.out.println(“Hello java”);
5. } 6. }
7. //compile by javac .java
8. //run by java A
```

compile it by **javac .java** run it by **java A**

9) Is delete,next,main,exit or null keyword in java?

No.

10) If I don’t provide any arguments on the command line, then the String array of Main method will be empty or null?

It is empty. But not null.

11) What if I write static public void instead of public static void?

Program compiles and runs properly.

12) What is the default value of the local variables?

The local variables are not initialized to any default value, neither primitives nor object references.

There is given more than 50 OOPs (Object-Oriented Programming and System) interview questions. But they have been categorized in many sections such as constructor interview questions, static interview questions, Inheritance Interview questions, Abstraction interview question, Polymorphism interview questions etc. for better understanding.

13) What is difference between object oriented programming language and object based programming language?

Object based programming languages follow all the features of OOPs except Inheritance. Examples of object based programming languages are JavaScript, VBScript etc.

14) What will be the initial value of an object reference which is defined as an instance variable?

The object references are all initialized to null in Java.

15) What is constructor?

• Constructor is just like a method that is used to initialize the state of an object. It is invoked at the time of object creation.

16) What is the purpose of default constructor?

• The default constructor provides the default values to the objects.

The java compiler creates a
default constructor only if there
is no constructor in the class.

17) Does constructor return any value?

Ans:yes, that is current instance (You cannot use return type yet it returns a value).

18)Is constructor inherited?

No, constructor is not inherited.

19) Can you make a constructor final?

No, constructor can’t be final.

20) What is static variable?

• static variable is used to refer the common property of all objects (that is not unique for each object) e.g. company name

of employees,college name of
students etc.

• static variable gets memory only once in class area at the time of
class loading.

21) What is static method?

• A static method belongs to the class rather than object of a
class.
• A static method can be invoked without the need for creating an
instance of a class.
• static method can access static data member and can change
the value of it.

22) Why main method is static?

because object is not required to call static method if

It were non-static method,jvm creats object first then call main() method that will lead to the problem of extra memory allocation.

23) What is static block?

• Is used to initialize the static
data member.
• It is executed before main method at the time of classloading.

24) Can we execute a program without main() method?

Ans) Yes, one of the way is static block.

25) What if the static modifier is removed from the signature of the main method?

Program compiles. But at runtime throws an error “NoSuchMethodError”.

26) What is difference between static (class) method and instance method?

static or class method
instance method
1)A method i.e. declared as static is known as static method.
A method i.e. not declared as static is known as instance method.

2)Object is not required to call static method. Object is required to call instance methods.

3)Non-static (instance) members cannot be accessed in static context (static method, static block and static nested class) directly.

static and non-static variables both can be accessed in instance methods.
4)For example: public static int cube(int n) { return n*n*n;}

For example: public void msg(){…}.
static context (static both can be
method, static block and static nested class) directly. accessed in instance methods.

4)For example: public static int cube(int n) { return n*n*n;}
For example: public void msg(){…}.

27) What is this in java?

It is a keyword that that refers to the current object.

28)What is Inheritance?

Inheritance is a mechanism in which one object acquires all the properties and behaviour of another object of another class. It represents IS-A relationship. It is used for Code Resusability and Method Overriding.

29) Which class is the superclass for every class.

Object class.

30) Why multiple inheritance is not supported in java?

• To reduce the complexity and simplify the language, multiple inheritance is not supported in
java in case of class.

31) What is composition?

Holding the reference of the other class within some other class is known as composition.

32) What is difference between aggregation and composition?

Aggregation represents weak relationship whereas composition represents strong relationship. For example: bike has an indicator (aggregation) but bike has an engine (compostion).

33) Why Java does not support pointers?

Pointer is a variable that refers to the memory address. They are not used in java because they are unsafe(unsecured) and complex to understand.

34) What is super in java?

It is a keyword that refers to the immediate parent class object.

35) Can you use this() and super() both in a constructor?

No. Because super() or this() must be the first statement.

36)What is object cloning?

The object cloning is used to create the exact copy of an object.

37) What is method overloading?

If a class have multiple methods by same name but different parameters, it is known as Method Overloading. It increases the readability of the program.

38) Why method overloading is not possible by changing the return type in java?

Because of ambiguity.

39) Can we overload main() method?

Yes, You can have many main() methods in a class by overloading the main method.

40) What is method overriding:

If a subclass provides a specific implementation of a method that is already provided by its parent class, it is known as Method Overriding. It is used for runtime polymorphism and to provide the specific implementation of the method.

41) Can we override static method?

No, you can't override the static method because they are the part of class not object.

42) Why we cannot override static method?

It is because the static method is the part of class and it is bound with class whereas instance method is bound with object and static gets memory in class area and instance gets memory in heap.

43) Can we override the overloaded method?

Yes.

44) Difference between method Overloading and Overriding.

Method Overriding

1) Method overloading increases the readability of the program.

Method overriding provides the specific implementation of the method that is already provided by its super class.

2) method overloading is occurs within the class. Method overriding occurs in two classes that have IS-A relationship.

3) In this In this case,
case, parameter must be parameter same.
must be
different.

overloading occurs in two classes
is occurs that have IS-A within the relationship. class. 3) In this case, parameter must be different. In this case, parameter must be same.

45) Can you have virtual functions in Java?

Yes, all functions in Java are virtual by default.

46) What is covariant return type?

Now, since java5, it is possible to override any method by changing the return type if the return type of the subclass overriding method is subclass type. It is known as

covariant return type.

47) What is final variable?

If you make any variable as final, you cannot change the value of final variable(It will be constant).

48) What is final method?

Final methods can’t be overriden.

49) What is final class?

Final class can’t be inherited.

50) What is blank final variable?

A final variable, not initalized at the time of declaration, is known as blank final variable.

18. Java Interview Questions part 2

Java Interview Questions

51) Can we intialize blank final variable?

Yes, only in constructor if it is non- static. If it is static blank final variable, it can be initialized only in the static block.

52) Can you declare the main method as final?

Yes, such as, public static final void main(String[] args){}.

53) What is Runtime Polymorphism?

Runtime polymorphism or dynamic method dispatch is a process in which a call to an overridden method is resolved at runtime rather than at compile-time. In this process, an overridden method

is called through the reference variable of a super class. The determination of the method to be called is based on the object being referred to by the reference variable.

54) Can you achieve Runtime Polymorphism by data members?

No.

55) What is the difference between static binding and dynamic binding?

In case of static binding type of object is determined at compile time whereas in dynamic binding type of object is determined at runtime.

56) What is abstraction?

Abstraction is a process of hiding the
implementation details and showing only functionality to the user.
Abstraction lets you focus on what the object does instead of how it does it.

57) What is the difference between abstraction and encapsulation?

Abstraction hides the implementation details whereas encapsulation wraps code and data into a single unit.

58) What is abstract class?

A class that is declared as abstract is known as abstract class. It needs to be extended and its method implemented. It cannot be instantiated.

59) Can there be any abstract method without abstract class?

No, if there is any abstract method in a class, that class must be abstract.

60) Can you use abstract and final both with a method?

No, because abstract method needs to be overridden whereas you can't override final method.

61) Is it possible to instantiate the abstract class?

No, abstract class can never be instantiated.

62) What is interface?

Interface is a blueprint of a class that have static constants and abstract methods.It can be used to achieve
fully abstraction and multiple inheritance.

63) Can you declare an interface method static?

No, because methods of an interface is abstract by default, and static and abstract keywords can't be used together .

64) Can an Interface be final?

No, because its implementation is provided by another class.

65) What is marker interface?

An interface that have no data member and method is known as a marker interface.For example Serializable, Cloneable etc.

66) What is difference between abstract class and interface?

Abstract class
Interface
1)An abstract class can have method body (nonabstract methods).

Interface have only abstract methods.
2)An abstract class can have instance variables. An interface cannot have instance variables. 3)An abstract class can have constructor . Interface cannot have constructor .
4)An abstract class can have static methods. Interface cannot have static methods.
5)You can extends one abstract class.
You can implement multiple interfaces.

67) Can we define private and protected modifiers for variables in interfaces?

No, they are implicitly public.

68) When can an object reference be cast to an interface reference?

An object reference can be cast to an interface reference when the object implements the referenced interface.

69) What is package?

A package is a group of similar type of classes interfaces and sub-packages. It provides access protection and removes naming collision.

70) Do I need to import java.lang package any time? Why ?

No. It is by default loaded internally by the JVM.

71) Can I import same package/class twice? Will the JVM load the package twice at runtime?

One can import the same package or same class multiple times. Neither compiler nor JVM complains about it.But the JVM will internally load the class only once no matter how many times you import the same class.

72) What is static import ?

By static import, we can access the static members of a class directly, there is no to qualify it with the class name.

73) What is Exception Handling?

Exception Handling is a mechanism to handle runtime errors.It is mainly used to handle checked exceptions.

74) What is difference between Checked Exception and Unchecked Exception? 1)Checked Exception

The classes that extend Throwable class except RuntimeException and Error are known as checked exceptions e.g.IOException,SQLException etc. Checked exceptions are checked at compile-time.

2)Unchecked Exception

The classes that extend RuntimeException are known as unchecked exceptions e.g. ArithmeticException,NullPointerExcept ion etc. Unchecked exceptions are not checked at compiletime.

75) What is the base class for Error and Exception?

Throwable.

76) Is it necessary that each try block must be followed by a catch block?

It is not necessary that each try block must be followed by a catch block. It should be followed by either a catch block OR a finally block. And whatever exceptions are likely to be thrown should be declared in the throws clause of the method.

77) What is finally block?

• finally block is a block that is always executed.

78) Can finally block be used without catch?

• Yes, by try block. finally must be
followed by either try or catch.

79) Is there any case when finally will not be executed?

finally block will not be executed if program exits(either by calling System.exit() or by causing a fatal error that causes the process to abort).

80) What is difference between throw and throws?

throw keyword
throws keyword

1)throw is used to explicitly throw an exception. throws is used to declare an exception.

2)checked exceptions can not be propagated with throw only.

checked exception can be propagated with throws. used to declare an

explicitly exception. throw an
exception.
2)checked exceptions can not be propagated with throw only.

checked exception can be propagated with throws. 3)throw is followed by an instance. throws is followed by class.

4)throw is used within the method.
throws is used with the method signature. 5)You cannot throw multiple exception

You can declare multiple exception e.g. public void method()throws IOException,SQLExc eption.

81) Can an exception be rethrown?

Yes.

82) Can subclass overriding method declare an exception if parent class method doesn't throw an exception ?

Yes but only unchecked exception not checked.

83) What is exception propagation ?

Forwarding the exception object to the invoking method is known as exception propagation.

There is given a list of string handling interview questions with short and pointed answers. If you know any string handling interview question, kindly post it in the comment section.

84) What is the meaning of immutable in terms of String?

The simple meaning of immutable is unmodifiable or unchangeable. Once string object has been created, its value can't be changed.

85) Why string objects are immutable in java?

Because java uses the concept of string literal. Suppose there are 5 reference variables,all referes to one object "sachin".If one reference variable changes the value of the object, it will be affected to all the reference variables. That is why string objects are immutable in java.

86) How many ways we can create the string object?

There are two ways to create the
string object, by string literal and by new keyword.

87) How many objects will be created in the following code?

1. String s1="Welcome";
2. String s2="Welcome";
3. String s3="Welcome"; Only one object.

88) Why java uses the concept of string literal?

To make Java more memory efficient (because no new objects are created if it exists already in string constant pool).

89)How many objects will be created in the following code?

1.
String s = **new** String("Welcome
");
Two objects, one in string constant pool and other in non-pool(heap).

90) What is the basic difference between string and stringbuffer object?

String is an immutable object. StringBuffer is a mutable object.

91) What is the difference between StringBuffer and StringBuilder ?

StringBuffer is synchronized whereas StringBuilder is not synchronized.

92) How can we create immutable class in java ?

We can create immutable class as the String class by defining final class and

93) What is the purpose of toString() method in java ?

The toString() method returns the string
representation of any object. If you print any object, java compiler internally invokes the toString() method on the object. So overriding the toString() method, returns the desired output, it can be the state of an object etc. depends on your implementation.

Core Java : Nested classes and Interfaces Interview Questions

94)What is nested class?

A class which is declared inside another class is known as nested class. There are 4 types of nested class member inner class, local inner class, annonymous inner class and static nested class.

95) Is there any difference between nested classes and inner classes?

Yes, inner classes are non-static nested classes i.e. inner classes are the part of nested classes.

96) Can we access the non-final local variable, inside the local inner class?

No, local variable must be constant if you want to access it in local inner class.

97) What is nested interface ?

Any interface i.e. declared inside the interface or class, is known as nested interface. It is static by default.

98) Can a class have an interface?

Yes, it is known as nested interface.

99) Can an Interface have a class?

Yes, they are static implicitely.

117) What is Garbage Collection?

Garbage collection is a process of
reclaiming the runtime unused
objects.It is performed for memory management.

118) What is gc()?

gc() is a daemon thread.gc() method is defined in System class that is used to send request to JVM to perform garbage collection.

119) What is the purpose of finalize() method?

finalize() method is invoked just before the object is garbage collected.It is used to perform cleanup processing.

120) Can an unrefrenced objects be refrenced again?

Yes.

121)What kind of thread is the Garbage collector thread?

Daemon thread.

122)What is difference between final, finally and finalize?

final: final is a keyword, final can be variable, method or class.You, can't change the value of final variable, can't override final method, can't inherit final class.

finally: finally block is used in exception handling. finally block is always executed.

finalize(): finalize() method is used in garbage collection.finalize() method is invoked just before the object is garbage collected.The finalize() method can be used to perform any cleanup processing.

123)What is the purpose of the Runtime class?

The purpose of the Runtime class is to provide access to the Java runtime system.

124)How will you invoke any external process in Java?

By Runtime.getRuntime().exec(?) method.

125)What is the difference between the Reader/Writer class hierarchy and the InputStream/OutputStream class hierarchy?

The Reader/Writer class hierarchy is characteroriented, and the InputStream/OutputStream class hierarchy is byte-oriented.

126)What an I/O filter?

An I/O filter is an object that reads from one stream and writes to another, usually altering the data in some way as it is passed from one

stream to another.

127) What is serialization?

Serialization is a process of writing the state of an object into a byte stream.It is mainly used to travel object's state on the network.

128) What is Deserialization?

Deserialization is the process of reconstructing the object from the serialized state.It is the reverse operation of serialization.

129) What is transient keyword?

If you define any data member as transient,it will not be serialized.

130)What is Externalizable?

Externalizable interface is used to write the state of an object into a byte stream in compressed format.It is not a marker interface.

131)What is the difference between Serializalble and Externalizable

interface?

Serializable is a marker interface but Externalizable is not a marker interface.When you use Serializable interface, your class is serialized automatically by default. But you can override writeObject() and readObject() two methods to control more complex object serailization process. When you use
Externalizable interface, you have a complete control over your class's serialization process.

132)How do I convert a numeric IP address like 192.18.97.39 into a hostname like java.sun.com?

By InetAddress.getByName("192.18.97.3 9").getHostName() where
192.18.97.39 is the IP address.

133) What is reflection?

Reflection is the process of examining or modifying the runtime behaviour of a class at runtime.It is used in:
• IDE (Integreted Development Environment) e.g. Eclipse,
MyEclipse, NetBeans.
• Debugger
• Test Tools etc.

134) Can you access the private method from outside the class?

Yes, by changing the runtime behaviour of a class if the class is not secured.