**Content**

|  |  |
| --- | --- |
| **S.NO** | **Topic** |
| 1 | Theory Notes |
| 2 | Local Variable |
| 3 | Unary Operators |
| 4 | Conditional Statements and Loops |
| 5 | Methods |

**Theory Notes**

**Path, -d, local variable, unary operators and methods.**

1. **What is path and why do we require path?**

Path is one of the environment variable. In order, to specify any command’s location to the command window path variable is useful.

1. **In how many ways we can set the path? Explain each.**

**Command window-wise:**

Using SET command provided by the OS.

SET path = location

Eg. SET path = D:\JDK8.0\bin;

**User-Wise:**

Right Click on My Computer 🡪 Properties 🡪 Select Advanced Tab 🡪 Click on Environment Variable button.

Under user level, click on new

Variable Name : PATH

Variable Value : location of bin folder.

Eg. Variable Name : PATH

Variable Value : D:\JDK8.0\bin

**All users of system(System-wise):**

Right Click on My Computer 🡪 Properties 🡪 Select Advanced Tab 🡪 Click on Environment Variable button.

Under system level, click on new

Variable Name : PATH

Variable Value : location of bin folder.

Eg. Variable Name : PATH

Variable Value : D:\JDK8.0\bin

[**Note :** environment variables and commands are case-insensitive.]

1. **How to read environment variable through command prompt?**

To read environment variable,

Echo %environmentVariable%

Eg. Echo %path%

1. **Specify the delimiters for environment variable value.**

In Windows, semicolon (;) is used as a delimiter.

Eg. To set path for java and maven

Set path = D:\JDK8.0\bin;D:\maven\apache-maven-3.2.3\bin

In Linux, colon (:) is used as a delimiter

Eg. To set path for java and maven

export PATH=/usr/local/JDK8.0/bin:/usr/local/apache-maven-3.1.1\bin

1. **What is the purpose of javac command?**

The purpose of javac command is to compile .java file. While compilation it checks whether the program is syntactically correct or not. If there is any mistake it will give compile-time error. Otherwise, it will generate .class file.

Eg. E:\Dev\app1\src>javac A.java

Assume A.java is syntactically correct. It will generate A.class file and place it in E:\Dev\app1\src folder.

1. **What is the purpose of java command?**

The purpose of java command is to run the .class file for specified classname.

While running the program it will start execution from main() method. If there is no main method it will give runtime error.

Eg: E:\Dev\app1\src> java A

Assume A contains main() method. It will show the output.

Eg: E:\Dev\app1\src> java B

(B is not containing main() method)

Runtime-error

Main method not found in class B, please define main method as :

public static void main(String[] args)

1. **Which are frequently used option with java and javac command?**

For javac command, -d (directory) is frequently used option. –d is used to specify the location to place class files which are generated by javac command.

Syntax : javac –d [location] [javafile]

location: where to place .class file

javafile: file to be compiled.

Eg.E:\Dev\app1\src>javac –d ../classes A.java

On executing above command it will compile A.java and generated A.class file will be placed in E:\Dev\app1\classes folder.

For java command, -cp (classpath) is frequently used option. –cp is used to specify location where java command should look for .class files to run. Through –cp option we are instructing java command to run .class file look in the specified directory.

Syntax: java –cp [location] [classname]

location: where to search for .class files

classname: name of class to run

1. **What is a variable?**

In order to store the varying data, then variable is required.

Syntax:variable declaration:datatype identifier;

datatype:type of data the variable can hold

identifier:name using which data will be identified.

Variables of same datatype can be declared in the same line using comma as a delimiter.

Syntax:datatype identifier1,identifier2,identifier3,..,identifier;

Eg. : int x1, count, sum, average;

Variable of same datatype can be initialized in the same line as under:

Syntax:identifier1=identifier2=identifier3=value;

Eg.: x1=count=sum=average=0;

Variable can hold only one value at any statement.

1. **How many things are associated with the variable? Specify each of them.**

There are 4 things associated with the variable

1. **Declaration**

datatype identifier;

Eg. int count;

1. **Initialization**

datatype identifier = initialValue;

Eg. int count = 0;

1. datatype identifier;

identifier = initialValue;

Eg. int count;

count = 0;

1. **Usage**

Variable can be used for various different

**purposes:**

1. Printing purpose

System.out.println(count);

1. Mathematical Operation

count = count + 10;

1. Provide value to another variable

int t = count;

1. Provide value for an argument in a method

class A

{

void sum(int c)

{

}

public static void main(String[] args)

{

int count = 10;

sum(count);

}

}

1. Hold value returned by a method.

class A

{

int sum(int c)

{

return ++c;

}

public static void main(String[] args)

{

int count = 10;

int x = sum(count);

}

}

1. Re-initialization

identifier = newValue;

x = 25;

1. **How many data types are there in Java?**

In Java, there are two datatypes :

1. Primitive Datatype
2. Derived Datatype
3. **What is Primitive Datatype? List out categories of Primitive Datatype.**

Primitive Datatypes are those defined in the programming language. These are also known as built-in datatypes.

1. For numbers without decimal point : byte, short, int, long
2. For numbers with decimal point : float, double
3. For true/false: boolean
4. For single character(any character from keyboard): char
5. **What is local variable? Specify rules to use local variable.**

Variable specified within a specific block such as method, constructor, instance initialization block, static initialization block, loop, condition block.

Local variables are not accessible outside block. Local variables are only visible to the methods in which they are declared; they are not accessible from the rest of the class.

Local variables must be declared and initialized before the first usage.

Local variables are not given default initial values. So it must be initialized before usage.

More than one local variable cannot have same name.

Local variables are created when the block is entered and destroyed once the block completes its execution.

Access modifiers such as static cannot be used with local variable.

Forward referencing is not allowed for local variables. As compilation as well as execution of block is from top to bottom.

If in a block, local variable is used without initialization then on compilation javac command gives error stating as

variable number might not have been initialized

When there is local and a global variable with the same name then local variable will have higher preference.

1. **What is unary operator? How many types of unary operator? How many types of usage?**

Unary operator is an operator which affects one unit to the variable value.

There are 2 types of unary operator:

1. Increment(++)

2. Decrement(--)

There are 2 types of usage:

1. **Pre-usage** : Changed value will be affecting from the same statement

**Pre-increment**: increment value by 1 and changed value will be affected from same statement.

**Pre-decrement:** decrement value by 1 and changed value will be affected from same statement.

1. **Post-usage:** Changed value will be affecting from the next immediate usage.

**Post-increment:** increment value by 1 and use new value in next immediate usage.

**Post-decrement:** decrement value by 1 and use new value in next immediate usage.

1. **What is a method?**

A method is a block containing statements to perform some operation.

The purpose of defining methods is reusability of the code.

A method should have a return type.

For using a method it should be called by default only main() method gets executed.

To execute methods other than main() calling it by providing appropriate arguments is necessary.

Method can have any number of arguments. Every argument should have a datatype and unique identifier.

Arguments are automatically initialized with the values from the calling statement so can be used without initialization in the method block. Arguments are local to the method.

A method can be also called as function, behavior.

1. **How many types of values a method can return? Name and specify necessary points to be considered for each of them.**

A method can return 3 different types of values:

1. void
2. Primitive
3. Derived

* **void:**

When method return type is void, return statement is optional.

If return statement is specified it should be without any value. i.e. return;

A call to method with void return type should be a statement.

ie. methodName();

Such methods cannot be used in mathematical operations or called within System.out.println().

* **Primitive/Derived:**

In methods with primitive/derived datatype as return type, return statement is compulsory.

If a return statement is specified it should return same type of value as one in method syntax.

A call to such methods can be as follows:

methodName();

ReturnType identifier = MethodName();

These methods can be used in mathematical operations or called within System.out.println().

return statement should always be last statement in a block.

**Conditional statements and Loops**

1. The only legal expression in an if statement is a boolean expression, in other words an expression that resolves to a boolean or a boolean variable.
2. Watch out for boolean assignments (=) that can be mistaken for boolean equality (==) tests:

boolean x = false;

if (x = true)

{

} //an assignment, so x will always be true!

1. Curly braces are optional for if blocks that have only one conditional statement. But watch out for misleading indentations.
2. switch statements can evaluate only to enums or the byte, short, int, and char data types. You can't say,

long s = 30;

switch(s) { }

1. The case constant must be a literal or final variable, or a constant expression, including an enum. You cannot have a case that includes a nonfinal variable, or a range of values.
2. If the condition in a switch statement matches a case constant, execution will run through all code in the switch following the matching case

statement until a break statement or the end of the switch statement is encountered. In other words, the matching case is just the entry point into the case block, but unless there's a break statement, the matching case is not the only case code that runs.

1. The default keyword should be used in a switch statement if you want to run some code when none of the case values match the conditional value.
2. The default block can be located anywhere in the switch block, so if no case matches, the default block will be entered, and if the default does not contain a break, then code will continue to execute (fall-through) to the end of the switch or until the break statement is encountered.
3. A basic for statement has three parts: declaration and/or initialization, Boolean evaluation, and the iteration expression.
4. If a variable is incremented or evaluated within a basic for loop, it must be declared before the loop, or within the for loop declaration.
5. A variable declared (not just initialized) within the basic for loop declaration cannot be accessed outside the for loop (in other words, code below the for loop won't be able to use the variable).
6. You can initialize more than one variable of the same type in the first part of the basic for loop declaration; each initialization must be separated by a comma.
7. An enhanced for statement (new as of Java 6), has two parts, the *declaration* and the *expression*. It is used only to loop through arrays or collections.
8. With an enhanced for, the *expression* is the array or collection through which you want to loop.
9. With an enhanced for, the *declaration* is the block variable, whose type is compatible with the elements of the array or collection, and that variable contains the value of the element for the given iteration.
10. You cannot use a number (old C-style language construct) or anything that does not evaluate to a boolean value as a condition for an if statement or looping construct. You can't, for example, say if(x), unless x is a Boolean variable.
11. The do loop will enter the body of the loop at least once, even if the test condition is not met.
12. An unlabeled break statement will cause the current iteration of the innermost looping construct to stop and the line of code following the loop to run.
13. An unlabeled continue statement will cause: the current iteration of the innermost loop to stop, the condition of that loop to be checked, and if the condition is met, the loop to run again.
14. If the break statement or the continue statement is labeled, it will cause similar action to occur on the labeled loop, not the innermost loop.

**Local variables and Unary Operators**

1. Program

class Hello

{

public static void main(String[] args)

{

System.out.println("Hello Hello");

}

}

Output🡪Hello Hello

1. Program

class HelloAgainAndAgain

{

public static void main(String[] args)

{

System.out.println("Hello World!");

}

}

Output🡪Hello World!

1. Program

class A

{

public static void main(String[] args)

{

System.out.println("Hello World!");

}

}

Output🡪Hello World!

1. Program

class B

{

public static void main(String[] args)

{

System.out.println("Hello World!");

System.out.println("Hello World!");

System.out.println("Hello World!");

}

}

Output🡪 Hello World!

Hello World!

Hello World!

1. Program

class C

{

public static void main(String[] args)

{

System.out.println("Hello World!");

System.out.println("Hello World!");

System.out.println("Hello World!");

System.out.println("Hello World!");

System.out.println("Hello World!");

System.out.println("Hello World!");

}

}

Output🡪Hello World!

Hello World!

Hello World!

Hello World!

Hello World!

Hello World!

1. Program

class D

{

public static void main(String[] args)

{

System.out.println("Hello World!");

System.out.println(10000);

System.out.println(10.989000);

System.out.println('a');

System.out.println('P');

System.out.println('%');

System.out.println(true);

System.out.println(false);

}

}

Output🡪Hello World!

10000

10.989

a

P

%

true

false

1. Program

class E

{

public static void main(String[] args)

{

System.out.println(10 + 20);

System.out.println(10 - 20);

System.out.println(10 / 20);

System.out.println(10 \* 20);

}

}

Output🡪30

-10

0

200

1. Program

class F

{

public static void main(String[] args)

{

System.out.println(100 == 100);

System.out.println(100 != 100);

}

}

Output🡪true

false

1. Program

class A

{

public static void main(String[] args)

{

int i = 0;

System.out.println(i);

}

}

Output🡪0

1. Program

class B

{

public static void main(String[] args)

{

double i = 10.9;

System.out.println(i);

}

}

Output🡪10.9

1. Program

class C

{

public static void main(String[] args)

{

int i = 12;

double j = 100.8;

char k = 'r';

boolean m = false;

String n = "hello";

System.out.println(i);

System.out.println(j);

System.out.println(k);

System.out.println(m);

System.out.println(n);

}

}

Output🡪12

100.8

r

false

hello

1. Program

class D

{

public static void main(String[] args)

{

int i = 100;

System.out.println(i);

i = 200;

System.out.println(i);

}

}

Output🡪100

200

1. Program

class E

{

public static void main(String[] args)

{

int i;

i = 10;

System.out.println(i);

i = 20;

System.out.println(i);

}

}

Output🡪10

20

1. Program

class F

{

public static void main(String[] args)

{

double j;

j = 0.8;

System.out.println(j);

j = 1.8;

System.out.println(j);

j = 100.9;

System.out.println(j);

}

}

Output🡪0.8

1.8

100.9

1. Program

class G

{

public static void main(String[] args)

{

int i = 10;

int j = 20;

System.out.println(i);

System.out.println(j);

}

}

Output🡪10

20

1. Program

class H

{

public static void main(String[] args)

{

int i, j = 20;

i = 10;

System.out.println(i);

System.out.println(j);

}

}

Output🡪10

20

1. Program

class I

{

public static void main(String[] args)

{

int x, y, z = 10, p;

x = 10;

y = p = 20;

System.out.println(x);

System.out.println(y);

System.out.println(z);

System.out.println(p);

}

}

Output🡪10

20

10

20

1. Program

class J

{

public static void main(String[] args)

{

double x, y = 10.9, z, m = 1.2;

x = 1.5;

z = x;

System.out.println(x);

System.out.println(y);

System.out.println(z);

System.out.println(m);

}

}

Output🡪1.5

10.9

1.5

1.2

1. Program

class K

{

public static void main(String[] args)

{

String s1, s2 = "xyz", s3 = "hello";

boolean f1 = true, f2, f3 = false;

s1 = "test";

f2 = true;

System.out.println(s1);

System.out.println(s2);

System.out.println(s3);

System.out.println(f1);

System.out.println(f2);

System.out.println(f3);

}

}

Output🡪test

xyz

hello

true

true

false

1. Program

class L

{

public static void main(String[] args)

{

int i = 10;

double i = 1.2;

System.out.println(i);

}

}

Output🡪compile time error

1. Program

class M

{

public static void main(String[] args)

{

int i = 10;

i = 20;

System.out.println(i);

}

}

Output🡪20

1. Program

class N

{

public static void main(String[] args)

{

int i;

System.out.println(i);

}

}

Output🡪compile time error

1. Program

class O

{

public static void main(String[] args)

{

int i;

int j = i;

System.out.println(i);

System.out.println(j);

}

}

Output🡪compile time error

1. Program

class P

{

public static void main(String[] args)

{

int i;

System.out.println(i = 10);

System.out.println(i);

}

}

Output🡪10

10

1. Program

class Q

{

public static void main(String[] args)

{

int i;

System.out.println(i = 10);

System.out.println(i = 20);

System.out.println(i = 30);

System.out.println(i);

}

}

Output🡪10

20

30

30

1. Program

class R

{

public static void main(String[] args)

{

String s1;

System.out.println(s1 = "hello");

System.out.println(s1);

}

}

Output🡪hello

hello

1. Program

class S

{

public static void main(String[] args)

{

int i;

int j = 10 + ( i = 20 );

System.out.println(i);

System.out.println(j);

}

}

Output🡪20

30

1. Program

class T

{

public static void main(String[] args)

{

int i;

int j = 10 + i;

System.out.println(i);

System.out.println(j);

}

}

Output🡪compile time error

1. Program

class U

{

public static void main(String[] args)

{

int i;

int j = 10 + ( i = 20 ) + i;

System.out.println(i);

System.out.println(j);

}

}

Output🡪20

50

1. Program

class V

{

public static void main(String[] args)

{

int i;

int j = 10 + i + (i = 30);

System.out.println(i);

System.out.println(j);

}

}

Output🡪compile time error

1. Program

class W

{

public static void main(String[] args)

{

int i;

System.out.println("done");

}

}

Output🡪done

1. Program

class A

{

public static void main(String[] args)

{

int i = 0;

System.out.println(i++);

System.out.println(i);

System.out.println(i);

}

}

Output🡪0

1

1

1. Program

class B

{

public static void main(String[] args)

{

int i = 0;

System.out.println(i--);

System.out.println(i);

System.out.println(i);

}

}

Output🡪0

-1

-1

1. Program

class C

{

public static void main(String[] args)

{

int i = 0;

int j = i++;

System.out.println(i);

System.out.println(j);

System.out.println(i);

System.out.println(j);

}

}

Output🡪1

0

1

0

1. Program

class D

{

public static void main(String[] args)

{

int i = 0;

int j = i--;

System.out.println(j);

System.out.println(i);

System.out.println(j);

System.out.println(i);

}

}

Output🡪0

-1

0

-1

1. Program

class E

{

public static void main(String[] args)

{

int i = 0;

int j = i++ + i;

System.out.println(i);

System.out.println(j);

}

}

Output🡪1

1

1. Program

class F

{

public static void main(String[] args)

{

int i = 0;

int j = i-- + i;

System.out.println(i);

System.out.println(j);

}

}

Output🡪-1

-1

1. Program

class G

{

public static void main(String[] args)

{

int i = 0;

int j = i++ + i + i++ + i;

System.out.println(i);

System.out.println(j);

}

}

Output🡪2

4

1. Program

class H

{

public static void main(String[] args)

{

int i = 0;

int j = i-- + i + i-- + i;

System.out.println(i);

System.out.println(j);

}

}

Output🡪-2

-4

1. Program

class I

{

public static void main(String[] args)

{

int x = 0;

int y = x++ + x++ + x++ + x;

System.out.println(x);

System.out.println(y);

}

}

Output🡪3

6

1. Program

class J

{

public static void main(String[] args)

{

int x = 0;

int y = x-- + x-- + x-- + x;

System.out.println(x);

System.out.println(y);

}

}

Output🡪-3

-6

1. Program

class K

{

public static void main(String[] args)

{

int i = 0;

int j = i++ + i + i-- + i;

System.out.println(i);

System.out.println(j);

}

}

Output🡪0

2

1. Program

class L

{

public static void main(String[] args)

{

int i = 0;

int j = i++ + i + i-- + i +

i-- + i++ + i-- + i +

i++ + i-- + i + i++;

System.out.println(i);

System.out.println(j);

}

}

Output🡪0

-3

1. Program

class M

{

public static void main(String[] args)

{

int i = 0;

System.out.println(++i);

System.out.println(i);

}

}

Output🡪1

1

1. Program

class N

{

public static void main(String[] args)

{

int i = 0;

System.out.println(--i);

System.out.println(i);

}

}

Output🡪-1

-1

1. Program

class O

{

public static void main(String[] args)

{

int i = 0;

int j = ++i;

System.out.println(i);

System.out.println(j);

}

}

Output🡪1

1

1. Program

class P

{

public static void main(String[] args)

{

int i = 0;

int j = --i;

System.out.println(i);

System.out.println(j);

}

}

Output🡪-1

-1

1. Program

class Q

{

public static void main(String[] args)

{

int i = 0;

int j = ++i + i;

System.out.println(i);

System.out.println(j);

}

}

Output🡪1

2

1. Program

class R

{

public static void main(String[] args)

{

int i = 0;

int j = --i + i;

System.out.println(i);

System.out.println(j);

}

}

Output🡪-1

-2

1. Program

class S

{

public static void main(String[] args)

{

int i = 0;

int j = ++i + ++i + ++i + i;

System.out.println(i);

System.out.println(j);

}

}

Output🡪3

9

1. Program

class T

{

public static void main(String[] args)

{

int i = 0;

int j = --i + --i + --i + i;

System.out.println(i);

System.out.println(j);

}

}

Output🡪-3

-9

1. Program

class U

{

public static void main(String[] args)

{

int i = 0;

int j = ++i + i + ++i + i;

i = 0;

int k = --i + i + --i + i;

System.out.println(i);

System.out.println(j);

System.out.println(k);

}

}

Output🡪-2

6

-6

1. Program

class V

{

public static void main(String[] args)

{

int i = 0;

int j = ++i + --i + ++i + i +

--i + i + --i + i +

++i + ++i + --i + i +

--i + i + --i + i;

System.out.println(i);

System.out.println(j);

}

}

Output🡪-2

**Loops and Conditional Statements**

1. Program

class A

{

public static void main(String[] args)

{

System.out.println("Hello World!");

System.out.println("Hello World!");

System.out.println("Hello World!");

System.out.println("Hello World!");

}

}

Output🡪Hello World!

Hello World!

Hello World!

Hello World!

1. Program

class B

{

public static void main(String[] args)

{

if(true)

{

System.out.println("from if");

}

System.out.println("end of main");

}

}

Output🡪from if

end of main

1. Program

class C

{

public static void main(String[] args)

{

if(false)

{

System.out.println("from if");

}

System.out.println("end of main");

}

}

Output🡪end of main

1. Program

class D

{

public static void main(String[] args)

{

int i = 10;

if(i == 10)

{

System.out.println("from if");

}

System.out.println("end of main");

}

}

Output🡪end of main

1. Program

class E

{

public static void main(String[] args)

{

int i = 10;

if(i != 10)

{

System.out.println("from if");

}

System.out.println("end of main");

}

}

Output🡪end of main

1. Program

class F

{

public static void main(String[] args)

{

int i = 10;

if(i = 10)

{

System.out.println("from if");

}

System.out.println("end of main");

}

}

Output🡪compile time error

1. Program

class G

{

public static void main(String[] args)

{

boolean flag = true;

if(flag)

{

System.out.println("from if");

}

System.out.println("end of main");

}

}

Output🡪from if

end of main

1. Program

class H

{

public static void main(String[] args)

{

boolean flag = false;

if(flag)

{

System.out.println("from if");

}

System.out.println("end of main");

}

}

Output🡪end of main

1. Program

class I

{

public static void main(String[] args)

{

boolean flag = true;

if(!flag)

{

System.out.println("from if");

}

System.out.println("end of main");

}

}

Output🡪end of main

1. Program

class J

{

public static void main(String[] args)

{

boolean flag = false;

if(!flag)

{

System.out.println("from if");

}

System.out.println("end of main");

System.out.println(flag);

}

}

Output🡪from if

end of main

false

1. Program

class K

{

public static void main(String[] args)

{

boolean flag = false;

if(flag = true)

{

System.out.println("from if");

}

System.out.println("end of main");

System.out.println(flag);

}

}

Output🡪from if

end of main

true

1. Program

class L

{

public static void main(String[] args)

{

boolean flag = true;

if(flag = false)

{

System.out.println("from if");

}

System.out.println("end of main");

System.out.println(flag);

}

}

Output🡪end of main

False

1. Program

class M

{

public static void main(String[] args)

{

if(!false)

{

System.out.println("from if");

}

System.out.println("end of main");

}

}

Output🡪from if

end of main

1. Program

class N

{

public static void main(String[] args)

{

boolean flag = true;

if(flag == true)

{

System.out.println("from if");

}

System.out.println("end of main");

System.out.println(flag);

}

}

Output🡪from if

end of main

true

1. Program

class O

{

public static void main(String[] args)

{

boolean flag = true;

if(flag == false)

{

System.out.println("from if");

}

System.out.println("end of main");

System.out.println(flag);

}

}

Output🡪end of main

true

1. Program

class P

{

public static void main(String[] args)

{

if(true && true)

{

System.out.println("from if");

}

System.out.println("end of main");

}

}

Output🡪from if

end of main

1. Program

class Q

{

public static void main(String[] args)

{

if(true && false)

{

System.out.println("from if");

}

System.out.println("end of main");

}

}

Output🡪end of main

1. Program

class R

{

public static void main(String[] args)

{

int i = 0;

if((i++ == 0) && (i++ == 1))

{

System.out.println("from if");

i++;

}

System.out.println("end of main");

System.out.println(i);

}

}

Output🡪from if

end of main

3

1. Program

class S

{

public static void main(String[] args)

{

int i = 0;

if((i++ == 1) && (i++ == 1))

{

System.out.println("from if");

i++;

}

System.out.println("end of main");

System.out.println(i);

}

}

Output🡪end of main

1

1. Program

class T

{

public static void main(String[] args)

{

int i = 0;

if((++i == 1) && (i++ == 2))

{

System.out.println("from if");

i++;

}

System.out.println("end of main");

System.out.println(i);

}

}

Output🡪end of main

2

1. Program

class U

{

public static void main(String[] args)

{

boolean flag = true;

if((flag = false) && (flag = true))

{

System.out.println("from if");

}

System.out.println("end of main");

System.out.println(flag);

}

}

Output🡪end of main

false

1. Program

class V

{

public static void main(String[] args)

{

if(false || true)

{

System.out.println("from if");

}

System.out.println("end of main");

}

}

Output🡪from if

end of main

1. Program

class W

{

public static void main(String[] args)

{

if(false || false)

{

System.out.println("from if");

}

System.out.println("end of main");

}

}

Output🡪end of main

1. Program

class X

{

public static void main(String[] args)

{

int i = 0;

if((i++ == 0) || (i++ == 1))

{

System.out.println("from if");

i++;

}

System.out.println("end of main");

System.out.println(i);

}

}

Output🡪from if

end of main

2

1. Program

class Y

{

public static void main(String[] args)

{

int i = 0;

if((i++ == 1) || (i++ == 1))

{

System.out.println("from if");

i++;

}

System.out.println("end of main");

System.out.println(i);

}

}

Output🡪from if

end of main

3

1. Program

class A

{

public static void main(String[] args)

{

if(true)

System.out.println(1);

System.out.println(2);

System.out.println(3);

}

}

Output🡪1

2

3

1. Program

class B

{

public static void main(String[] args)

{

if(false)

System.out.println(1);

System.out.println(2);

System.out.println(3);

}

}

Output🡪2

3

1. Program

class C

{

public static void main(String[] args)

{

if(false);

System.out.println(1);

System.out.println(2);

System.out.println(3);

}

}

Output🡪1

2

3

1. Program

class D

{

public static void main(String[] args)

{

if(true)

{

System.out.println("from if");

}

else

{

System.out.println("from else");

}

System.out.println("main end");

}

}

Output🡪from if

main end

1. Program

class E

{

public static void main(String[] args)

{

if(false)

{

System.out.println("from if");

}

else

{

System.out.println("from else");

}

System.out.println("main end");

}

}

Output🡪from else

main end

1. Program

class F

{

public static void main(String[] args)

{

else

{

System.out.println("done");

}

}

}

Output🡪compile time error

1. Program

class G

{

public static void main(String[] args)

{

if(true)

{

System.out.println("Hello World!");

}

System.out.println("Hello World!");

else

{

System.out.println("Hello World!");

}

}

}

Output🡪compile time error

1. Program

class H

{

public static void main(String[] args)

{

if(true)

{

System.out.println("if");

if(true)

{

System.out.println("inner if");

}

}

}

}

Output🡪if

inner if

1. Program

class I

{

public static void main(String[] args)

{

if(true)

if(true)

System.out.println("inner if");

else

System.out.println("else1");

else

System.out.println("else2");

}

}

Output🡪inner if

1. Program

class J

{

public static void main(String[] args)

{

if(false)

if(true)

System.out.println("inner if");

else

System.out.println("else1");

else

System.out.println("else2");

}

}

Output🡪else2

1. Program

class K

{

public static void main(String[] args)

{

if(true)

if(false)

System.out.println("inner if");

else

System.out.println("else1");

else

System.out.println("else2");

}

}

Output🡪else1

1. Program

class L

{

public static void main(String[] args)

{

if(true)

{

System.out.println("from if");

}

else if(true)

{

System.out.println("from else if");

}

else if(true)

{

System.out.println("from last else if");

}

}

}

Output🡪from if

1. Program

class M

{

public static void main(String[] args)

{

if(false)

{

System.out.println("from if");

}

else if(true)

{

System.out.println("from else if");

}

else if(true)

{

System.out.println("from last else if");

}

}

}

Output🡪from else if

1. Program

class N

{

public static void main(String[] args)

{

if(false)

{

System.out.println("from if");

}

else if(false)

{

System.out.println("from else if");

}

else if(true)

{

System.out.println("from last else if");

}

}

}

Output🡪from last else if

1. Program

class O

{

public static void main(String[] args)

{

if(false)

{

System.out.println("from if");

}

else if(false)

{

System.out.println("from else if");

}

else if(false)

{

System.out.println("from last else if");

}

}

}

Output🡪no output

1. Program

class P

{

public static void main(String[] args)

{

if(false)

{

System.out.println("from if");

}

else if(false)

{

System.out.println("from else if1");

}

else if(false)

{

System.out.println("from else if2");

}

else if(false)

{

System.out.println("from else if3");

}

else

{

System.out.println("from last else");

}

}

}

Output🡪from last else

1. Program

class Q

{

public static void main(String[] args)

{

int i = 2;

if(i > 10)

{

System.out.println("i > 10");

}

else if(i > 5)

{

System.out.println("i > 5");

}

else if(i > 2)

{

System.out.println("i > 2");

}

else

{

System.out.println("nothing");

}

}

}

Output🡪from last else

1. Program

class A

{

public static void main(String[] args)

{

for(int i = 0; i < 10; i++)

{

System.out.println(i);

}

}

}

Output🡪0

1

2

3

4

5

6

7

8

9

1. Program

class C

{

public static void main(String[] args)

{

for(int i = 0; i > 0; i++)

{

System.out.println("loop:" + i);

}

System.out.println("main end");

}

}

Output🡪main end

1. Program

class D

{

public static void main(String[] args)

{

for(int i = 0; i < 10; i++)

{

System.out.println("loop:" + i);

}

System.out.println("main:" + i);

}

}

Output🡪compile time error

1. Program

class E

{

public static void main(String[] args)

{

int i;

for(i = 0; i < 10; i++)

{

System.out.println("loop:" + i);

}

System.out.println("main:" + i);

}

}

Output🡪loop:0

loop:1

loop:2

loop:3

loop:4

loop:5

loop:6

loop:7

loop:8

loop:9

main:10

1. Program

class F

{

public static void main(String[] args)

{

for(int i = 0; i < 10; i++)

System.out.println("loop:" + i);

System.out.println("main");

}

}

Output🡪loop:0

loop:1

loop:2

loop:3

loop:4

loop:5

loop:6

loop:7

loop:8

loop:9

main

1. Program

class G

{

public static void main(String[] args)

{

for(int i = 0; i < 10; i++);

System.out.println("loop:" + i);

System.out.println("main");

}

}

Output🡪compile time error

1. Program

class H

{

public static void main(String[] args)

{

int i;

for(i = 0; i < 5; i++);

System.out.println("loop:" + i);

System.out.println("main:" + i);

}

}

Output🡪loop:5

main:5

1. Program

class I

{

public static void main(String[] args)

{

for(int i = 5; i > 0; i--)

System.out.println("loop:" + i);

System.out.println("main:" + i);

}

}

Output🡪compile time error

1. Program

class J

{

public static void main(String[] args)

{

int i;

for(i = 5; i > 0; i--)

System.out.println("loop:" + i);

System.out.println("main:" + i);

}

}

Output🡪loop:5

loop:4

loop:3

loop:2

loop:1

main:0

1. Program

class K

{

public static void main(String[] args)

{

for(int i = 0, j = 10; i < 5; i++, j--)

{

System.out.println(i + ":" + j);

}

}

}

Output🡪0:10

1:9

2:8

3:7

4:6

1. Program

class L

{

public static void main(String[] args)

{

for(int i = 0, j = 10; j > 2; j--, i++)

{

System.out.println(i + ":" + j);

}

}

}

Outout🡪0:10

1:9

2:8

3:7

4:6

5:5

6:4

7:3

1. Program

class M

{

public static void main(String[] args)

{

for(int i = 0, j = 20;

(i < 15 || j > 15);

i++, j--)

{

System.out.println(i + "," + j);

}

}

}

Output🡪0,20

1,19

2,18

3,17

4,16

5,15

6,14

7,13

8,12

9,11

10,10

11,9

12,8

13,7

14,6

1. Program

class N

{

public static void main(String[] args)

{

for(int i = 0, j = 20;

(i < 15 && j > 15);

i++, j--)

{

System.out.println(i + "," + j);

}

}

}

Output🡪0,20

1,19

2,18

3,17

4,16

1. Program

class O

{

public static void main(String[] args)

{

int i;

for(i = 0, System.out.println("INITIALIZER");

i < 5;

i++, System.out.println("INCREMENT"))

{

System.out.println("BODY");

}

}

}

Output🡪INITIALIZER

BODY

INCREMENT

BODY

INCREMENT

BODY

INCREMENT

BODY

INCREMENT

BODY

INCREMENT

1. Program

class P

{

public static void main(String[] args)

{

int i;

for(i = 0, System.out.println("INITIALIZER");

i < 5;

i++, System.out.println("INCREMENT"))

{

System.out.println("BODY");

}

}

}

Output🡪INITIALIZER

BODY

INCREMENT

BODY

INCREMENT

BODY

INCREMENT

BODY

INCREMENT

BODY

INCREMENT

1. Program

class Q

{

public static void main(String[] args)

{

for(int i = 0; i < 10; i++)

{

System.out.println("loop begin");

if(i > 5)

{

continue;

}

System.out.println("loop end");

}

System.out.println("main end");

}

}

Output🡪loop begin

loop end

loop begin

loop end

loop begin

loop end

loop begin

loop end

loop begin

loop end

loop begin

loop end

loop begin

loop begin

loop begin

loop begin

main end

1. Program

class R

{

public static void main(String[] args)

{

for(int i = 0; i < 10; i++)

{

System.out.println("loop begin");

if(i > 5)

{

break;

}

System.out.println("loop end");

}

System.out.println("main end");

}

}

Output🡪loop begin

loop end

loop begin

loop end

loop begin

loop end

loop begin

loop end

loop begin

loop end

loop begin

loop end

loop begin

main end

1. Program

class S

{

public static void main(String[] args)

{

for(int i = 0; i < 10; i++)

{

System.out.println("loop begin");

if(i > 5)

{

break;

System.out.println("if block");

}

System.out.println("loop end");

}

System.out.println("main end");

}

}

Output🡪compile time error

1. Program

class T

{

public static void main(String[] args)

{

for(int i = 0; i < 2; i++)

{

System.out.println("loop1 begin");

for(int j = 0; j < 5; j++)

{

System.out.println("loop2 begin");

if(j > 2)

{

continue;

}

System.out.println("loop2 end");

}

System.out.println("loop1 end");

}

System.out.println("main end");

}

}

Output🡪loop1 begin

loop2 begin

loop2 end

loop2 begin

loop2 end

loop2 begin

loop2 end

loop2 begin

loop2 begin

loop1 end

loop1 begin

loop2 begin

loop2 end

loop2 begin

loop2 end

loop2 begin

loop2 end

loop2 begin

loop2 begin

loop1 end

main end

1. Program

class U

{

public static void main(String[] args)

{

for(int i = 0; i < 2; i++)

{

System.out.println("loop1 begin");

for(int j = 0; j < 5; j++)

{

System.out.println("loop2 begin");

if(j > 2)

{

break;

}

System.out.println("loop2 end");

}

System.out.println("loop1 end");

}

System.out.println("main end");

}

}

Output🡪loop1 begin

loop2 begin

loop2 end

loop2 begin

loop2 end

loop2 begin

loop2 end

loop2 begin

loop1 end

loop1 begin

loop2 begin

loop2 end

loop2 begin

loop2 end

loop2 begin

loop2 end

loop2 begin

loop1 end

main end

1. Program

class V

{

public static void main(String[] args)

{

loop1:

for(int i = 0; i < 2; i++)

{

System.out.println("loop1 begin");

for(int j = 0; j < 5; j++)

{

System.out.println("loop2 begin");

if(j > 2)

{

continue loop1;

}

System.out.println("loop2 end");

}

System.out.println("loop1 end");

}

}

}

Output🡪loop1 begin

loop2 begin

loop2 end

loop2 begin

loop2 end

loop2 begin

loop2 end

loop2 begin

loop1 begin

loop2 begin

loop2 end

loop2 begin

loop2 end

loop2 begin

loop2 end

loop2 begin

1. Program

class W

{

public static void main(String[] args)

{

loop1:

for(int i = 0; i < 2; i++)

{

System.out.println("loop1 begin");

for(int j = 0; j < 5; j++)

{

System.out.println("loop2 begin");

if(j > 2)

{

break loop1;

}

System.out.println("loop2 end");

}

System.out.println("loop1 end");

}

}

}

Output🡪loop1 begin

loop2 begin

loop2 end

loop2 begin

loop2 end

loop2 begin

loop2 end

loop2 begin

1. Program

class X

{

public static void main(String[] args)

{

int[] x = {10, 20, 12, 5, 26};

for(int i : x) // for-each

{

System.out.println(i);

}

}

}

Output🡪10

20

12

5

26

1. Program

class Y

{

public static void main(String[] args)

{

double[] x = {10.4, 20.1, 12.4, 5.7};

for(double i : x) // for-each

{

System.out.println(i);

}

}

}

Output🡪10.4

20.1

12.4

5.7

1. Program

class Z

{

public static void main(String[] args)

{

String[] str = {"abc", "xyz", "test"};

for(String s1 : str)

{

System.out.println(s1);

}

}

}

Output🡪abc

xyz

test

1. Program

class Z1

{

public static void main(String[] args)

{

boolean[] x = {false, true, false};

for(int i : x)

{

System.out.println(i);

}

}

}

Output🡪compile time error

1. Program

class Z2

{

public static void main(String[] args)

{

int[] x = {9, 7, 4, 10};

int i;

for(i : x)

{

System.out.println(i);

}

}

}

Output🡪compile time error

1. Program

class A

{

public static void main(String[] args)

{

int i = 0;

while(i < 5)

{

System.out.println(i);

i++;

}

}

}

Output🡪0

1

2

3

4

1. Program

class B

{

public static void main(String[] args)

{

int i = 5;

while(i > 0)

{

System.out.println(i);

i--;

}

}

}

Output🡪5

4

3

2

1

1. Program

class C

{

public static void main(String[] args)

{

int i = 0;

while((i = 2) < 5)

{

System.out.println(i);

i++;

}

}

}

Output🡪2

2

2

!

!

(2 infinite time)

1. Program

class D

{

public static void main(String[] args)

{

int i = 0;

while(i < 5)

{

i++;

}

System.out.println(i);

}

}

Output🡪5

1. Program

class E

{

public static void main(String[] args)

{

int i = 0;

while(i < 5);

{

i++;

}

System.out.println(i);

}

}

Output🡪running success (no output)

1. Program

class F

{

public static void main(String[] args)

{

int i = 0;

while(i < 4)

{

System.out.println("begin");

if(i > 2)

{

continue;

}

System.out.println("end");

i++;

}

}

}

Output🡪begin

begin

!

!

(begin infinite time)

1. Program

class G

{

public static void main(String[] args)

{

int i = 0;

while(i < 4)

{

i++;

System.out.println("begin");

if(i > 2)

{

continue;

}

System.out.println("end");

}

}

}

Output🡪begin

end

begin

end

begin

begin

1. Program

class H

{

public static void main(String[] args)

{

int i = 0;

while(i < 5)

{

System.out.println("begin");

if(i > 2)

{

break;

}

System.out.println("end");

i++;

}

}

}

Output🡪begin

end

begin

end

begin

end

begin

1. Program

class I

{

public static void main(String[] args)

{

int i = 0;

while(i < 5)

{

System.out.println("loop1 begin");

int j = 0;

while(j < 5)

{

j++;

System.out.println("loop2 begin");

if(j > 1)

{

continue;

}

System.out.println("loop2 end");

}

System.out.println("loop1 end");

i++;

}

}

}

Output🡪loop1 begin

loop2 begin

loop2 end

loop2 begin

loop2 begin

loop2 begin

loop2 begin

loop1 end

loop1 begin

loop2 begin

loop2 end

loop2 begin

loop2 begin

loop2 begin

loop2 begin

loop1 end

loop1 begin

loop2 begin

loop2 end

loop2 begin

loop2 begin

loop2 begin

loop2 begin

loop1 end

loop1 begin

loop2 begin

loop2 end

loop2 begin

loop2 begin

loop2 begin

loop2 begin

loop1 end

loop1 begin

loop2 begin

loop2 end

loop2 begin

loop2 begin

loop2 begin

loop2 begin

loop1 end

1. Program

class J

{

public static void main(String[] args)

{

int i = 0;

while(i < 5)

{

System.out.println("loop1 begin");

int j = 0;

while(j < 5)

{

j++;

System.out.println("loop2 begin");

if(j > 1)

{

break;

}

System.out.println("loop2 end");

}

System.out.println("loop1 end");

i++;

}

}

}

Output🡪loop1 begin

loop2 begin

loop2 end

loop2 begin

loop1 end

loop1 begin

loop2 begin

loop2 end

loop2 begin

loop1 end

loop1 begin

loop2 begin

loop2 end

loop2 begin

loop1 end

loop1 begin

loop2 begin

loop2 end

loop2 begin

loop1 end

loop1 begin

loop2 begin

loop2 end

loop2 begin

loop1 end

1. Program

class K

{

public static void main(String[] args)

{

System.out.println("main begin");

int i = 0;

do

{

System.out.println("body:" + i);

i++;

}

while (i < 5);

System.out.println("main end");

}

}

Output🡪main begin

body:0

body:1

body:2

body:3

body:4

main end

1. Program

class L

{

public static void main(String[] args)

{

int i = 10;

do

{

System.out.println("body:" + i);

i++;

}

while (i < 10);

}

}

Output🡪body:10

1. Program

class M

{

public static void main(String[] args)

{

int i = 0;

do

{

System.out.println("begin");

i++;

if(i > 3)

{

continue;

}

System.out.println("end");

}

while (i < 10);

}

}

Output🡪begin

end

begin

end

begin

end

begin

begin

begin

begin

begin

begin

begin

1. Program

class N

{

public static void main(String[] args)

{

int i = 0;

do

{

System.out.println("begin");

i++;

if(i > 3)

{

break;

}

System.out.println("end");

}

while (i < 10);

}

}

Output🡪begin

end

begin

end

begin

end

begin

1. Program

class O

{

public static void main(String[] args)

{

int i = 0;

do

{

System.out.println("loop1 begin:" + i);

int j = 0;

do

{

System.out.println("loop2 begin:" + j);

j++;

System.out.println("loop2 end:" + j);

}

while (j < 2);

i++;

System.out.println("loop1 end:" + i);

}

while (i < 3);

}

}

Output🡪loop1 begin:0

loop2 begin:0

loop2 end:1

loop2 begin:1

loop2 end:2

loop1 end:1

loop1 begin:1

loop2 begin:0

loop2 end:1

loop2 begin:1

loop2 end:2

loop1 end:2

loop1 begin:2

loop2 begin:0

loop2 end:1

loop2 begin:1

loop2 end:2

loop1 end:3

1. Program

class P

{

public static void main(String[] args)

{

int i = 0;

do

{

System.out.println("loop1 begin:" + i);

int j = 0;

do

{

System.out.println("loop2 begin:" + j);

j++;

if(j > 1)

{

continue;

}

System.out.println("loop2 end:" + j);

}

while (j < 2);

i++;

System.out.println("loop1 end:" + i);

}

while (i < 3);

}

}

Output🡪loop1 begin:0

loop2 begin:0

loop2 end:1

loop2 begin:1

loop1 end:1

loop1 begin:1

loop2 begin:0

loop2 end:1

loop2 begin:1

loop1 end:2

loop1 begin:2

loop2 begin:0

loop2 end:1

loop2 begin:1

loop1 end:3

1. Program

class Q

{

public static void main(String[] args)

{

int i = 3;

switch(i)

{

case 2:

System.out.println("from 2");

case 5:

System.out.println("from 5");

case 3:

System.out.println("from 3");

}

}

}

Output🡪from 3

1. Program

class R

{

public static void main(String[] args)

{

int i = 3;

switch(i)

{

case 2:

System.out.println("from 2");

case 3:

System.out.println("from 3");

case 5:

System.out.println("from 5");

}

}

}

Output🡪from 3

from 5

1. Program

class S

{

public static void main(String[] args)

{

int i = 3;

switch(i)

{

case 2:

System.out.println("from 2");

break;

case 3:

System.out.println("from 3");

break;

case 5:

System.out.println("from 5");

break;

}

}

}

Output🡪from 3

1. Program

class T

{

public static void main(String[] args)

{

int i = 3;

switch(i)

{

case 2:

System.out.println("from 2");

System.out.println("from 2");

System.out.println("from 2");

break;

case 3:

System.out.println("from 3");

System.out.println("from 3");

System.out.println("from 3");

break;

case 5:

System.out.println("from 5");

System.out.println("from 5");

System.out.println("from 5");

break;

}

}

}

Output🡪from 3

from 3

from 3

1. Program

class U

{

public static void main(String[] args)

{

int i = 10;

switch(i)

{

case 5:

System.out.println("from 5");

case 3:

System.out.println("from 3");

case 20:

System.out.println("from 20");

}

System.out.println("main end");

}

}

Output🡪main end

1. Program

class V

{

public static void main(String[] args)

{

int i = 10;

switch(i)

{

case 5:

System.out.println("from 5");

case 3:

System.out.println("from 3");

case 20:

System.out.println("from 20");

default:

System.out.println("from default");

}

System.out.println("main end");

}

}

Output:from default

main end

1. Program

class W

{

public static void main(String[] args)

{

int i = 10;

switch(i)

{

case 5:

System.out.println("from 5");

default:

System.out.println("from default");

case 3:

System.out.println("from 3");

case 20:

System.out.println("from 20");

}

System.out.println("main end");

}

}

Output: from default

from 3

from 20

main end

1. Program

class X

{

public static void main(String[] args)

{

int i = 10;

switch(i)

{

case 5:

System.out.println("from 5");

default:

System.out.println("from default");

break;

case 3:

System.out.println("from 3");

case 20:

System.out.println("from 20");

}

System.out.println("main end");

}

}

Output:from default

main end

1. Program

class Y

{

public static void main(String[] args)

{

int i = 11;

switch(i)

{

case 2:

case 4:

case 6:

case 8:

case 10:

System.out.println("even");

break;

case 1:

case 3:

case 5:

case 7:

System.out.println("odd");

break;

default:

System.out.println("num is greater than 10");

}

}

}

Output: num is greater than 10

1. Program

class B

{

public static void main(String[] args)

{

for(int i = 0; i < 10; i++)

{

for(int j = 0; j < 3; j++)

{

System.out.print(i);

System.out.print(j);

}

System.out.println();

}

}

}

Output: 000102

101112

202122

303132

404142

505152

606162

707172

808182

909192

***Methods***

1. Program

class A

{

public static void main(String[] args)

{

System.out.println("from main");

}

public static void test()

{

System.out.println("from test");

}

}

Output: from main

1. Program

class B

{

public static void test(String[] args)

{

System.out.println("from test");

}

public static void main(String[] args)

{

System.out.println("from main");

}

}

Output: from main

1. Program

class C

{

public static void test1()

{

System.out.println("from test1");

}

public static void main(String[] args)

{

System.out.println("from main");

}

public static void test2()

{

System.out.println("from test2");

}

}

Output: from main

1. Program

class D

{

public static void test()

{

System.out.println("from test");

}

public static void main(String[] args)

{

System.out.println("main begin");

test();

System.out.println("main end");

}

}

Output: main begin

from test

main end

1. Program

class E

{

public static void main(String[] args)

{

System.out.println("main begin");

test1();

System.out.println("main end");

}

public static void test1()

{

System.out.println("from test1");

System.out.println("from test1");

System.out.println("from test1");

}

}

Output: main begin

from test1

from test1

from test1

main end

1. Program

class F

{

public static void main(String[] args)

{

System.out.println("main begin");

test1();

System.out.println("----------");

test1();

System.out.println("main end");

}

public static void test1()

{

System.out.println("from test1");

}

}

Output: main begin

from test1

----------

from test1

main end

1. Program

class G

{

public static void main(String[] args)

{

System.out.println("main begin");

test1();

System.out.println("main end");

}

public static void test1()

{

System.out.println("test1 begin");

test2();

System.out.println("test1 end");

}

public static void test2()

{

System.out.println("test2 begin");

System.out.println("test2 end");

}

}

Output: main begin

test1 begin

test2 begin

test2 end

test1 end

main end

1. Program

class H

{

public static void main(String[] args)

{

System.out.println("main begin");//1

test1();//2

test2();//4

System.out.println("main end");//6

}

public static void test1()

{

System.out.println("from test1");//3

}

public static void test2()

{

System.out.println("from test2");//5

}

}

Output: main begin

from test1

from test2

main end

1. Program

class I

{

public static void main(String[] args)

{

System.out.println("main begin");

test();

System.out.println("main end");

}

static void test()

{

System.out.println("from test");

}

}

Output: main begin

from test

main end

1. Program

class J

{

public static void main(String[] args)

{

System.out.println("main begin");

test();

System.out.println("main end");

}

static void test()

{

System.out.println("from test");

return;

}

}

Output🡪main begin

from test

main end

1. Program

class K

{

public static void main(String[] args)

{

System.out.println("from main");

return;

}

}

Output🡪from main

1. Program

class L

{

public static void main(String[] args)

{

System.out.println("from main");

return 100;

}

}

Output🡪compile time error

1. Program

class M

{

public static void main(String[] args)

{

System.out.println("main begin");

return;

System.out.println("main end");

}

}

Output🡪compile time error

1. Program

class N

{

public static void main(String[] args)

{

System.out.println("main begin");

if(true)

{

System.out.println("from if");

return;

}

System.out.println("main end");

}

}

Output🡪main begin

from if

1. Program

class O

{

public static void test()

{

System.out.println("test begin");

if(true)

{

System.out.println("from if");

return;

}

System.out.println("test end");

}

public static void main(String[] args)

{

System.out.println("main begin");

test();

System.out.println("main end");

}

}

Output🡪main begin

test begin

from if

main end

1. Program

class P

{

public static void main(String[] args)

{

System.out.println("byte min:" + Byte.MIN\_VALUE);

System.out.println("byte max:" + Byte.MAX\_VALUE);

System.out.println();

System.out.println("Short min:" + Short.MIN\_VALUE);

System.out.println("Short max:" + Short.MAX\_VALUE);

System.out.println();

System.out.println("int min:" + Integer.MIN\_VALUE);

System.out.println("int max:" + Integer.MAX\_VALUE);

System.out.println();

System.out.println("long min:" + Long.MIN\_VALUE);

System.out.println("long max:" + Long.MAX\_VALUE);

System.out.println();

System.out.println("float min:" + Float.MIN\_VALUE);

System.out.println("float max:" + Float.MAX\_VALUE);

System.out.println();

System.out.println("double min:" + Double.MIN\_VALUE);

System.out.println("double max:" + Double.MAX\_VALUE);

}

}

Output🡪 byte min:-128

byte max:127

Short min:-32768

Short max:32767

int min:-2147483648

int max:2147483647

long min:-9223372036854775808

long max:9223372036854775807

float min:1.4E-45

float max:3.4028235E38

double min:4.9E-324

double max:1.7976931348623157E308

/\*

Primitive datatypes

byte, short, int, long

float, double

boolean

char

\*/

1. Program

class Q

{

public static int test()

{

System.out.println("from test");

}

public static void main(String[] args)

{

System.out.println("from main");

}

}

Output🡪Compile time error

1. Program

class R

{

public static int test()

{

System.out.println("from test");

return 200;

}

public static void main(String[] args)

{

System.out.println("from main");

}

}

Output🡪from main

1. Program

class S

{

public static int test()

{

System.out.println("from test");

return 200;

}

public static void main(String[] args)

{

System.out.println("from main");

int i = test();

System.out.println("i = " + i);

}

}

Output🡪from main

from test

i = 200

1. Program

class T

{

public static int test()

{

System.out.println("from test");

return 200;

}

public static void main(String[] args)

{

System.out.println("from main");

int i = test();

int j = i + test();

System.out.println("i = " + i);

System.out.println("j = " + j);

}

}

Output: from main

from test

from test

i = 200

j = 400

1. Program

class U

{

public static int test()

{

System.out.println("from test");

return 200;

}

public static void main(String[] args)

{

System.out.println("from main");

int i = test();

int j = i + test();

int k = i + test() + j + test();

System.out.println("i = " + i);

System.out.println("j = " + j);

System.out.println("k = " + k);

System.out.println(test());

}

}

Output: from main

from test

from test

from test

from test

i = 200

j = 400

k = 1000

from test

200

1. Program

class V

{

public static String test()

{

System.out.println("from test");

}

public static void main(String[] args)

{

System.out.println("from main");

}

}

Output: V.java:6: error: missing return statement

}

^

1 error

1. Program

class W

{

public static String test()

{

System.out.println("from test");

return "abc";

}

public static void main(String[] args)

{

System.out.println("from main");

}

}

Output: from main

1. Program

class X

{

public static String test()

{

System.out.println("from test");

return "abc";

}

public static void main(String[] args)

{

System.out.println("from main");

String s1 = test();

System.out.println("s1:" + s1);

System.out.println(test());

}

}

Output: from main

from test

s1:abc

from test

abc

1. Program

class Y

{

public static void test(int i)

{

System.out.println("from test:" + i);

}

public static void main(String[] args)

{

System.out.println("from main");

test();

}

}

Output:Y.java:10: error: method test in class A cannot be applied to given types;

test();

^

required: int

found: no arguments

reason: actual and formal argument lists differ in length

1 error

1. Program

class Z

{

public static void test(int i)

{

System.out.println("from test:" + i);

}

public static void main(String[] args)

{

System.out.println("from main");

test(200);

}

}

Output: from main

from test:200

1. Program

class Z1

{

public static void test(int i, char j)

{

System.out.println("from test:" + i);

System.out.println("from test:" + j);

}

public static void main(String[] args)

{

System.out.println("from main");

test(200, 'z');

}

}

Output: from main

from test:200

from test:z

1. Program

class Z2

{

static double test(boolean b,

int i,

char j,

int k)

{

System.out.println("from test");

System.out.println(i);

System.out.println(j);

System.out.println(k);

System.out.println(b);

return 500.909;

}

public static void main(String[] args)

{

System.out.println("from main");

double d1 = test(true, 10, 'a', 30);

System.out.println("from main:" + d1);

}

}

Output: from main

from test

10

a

30

true

from main:500.909