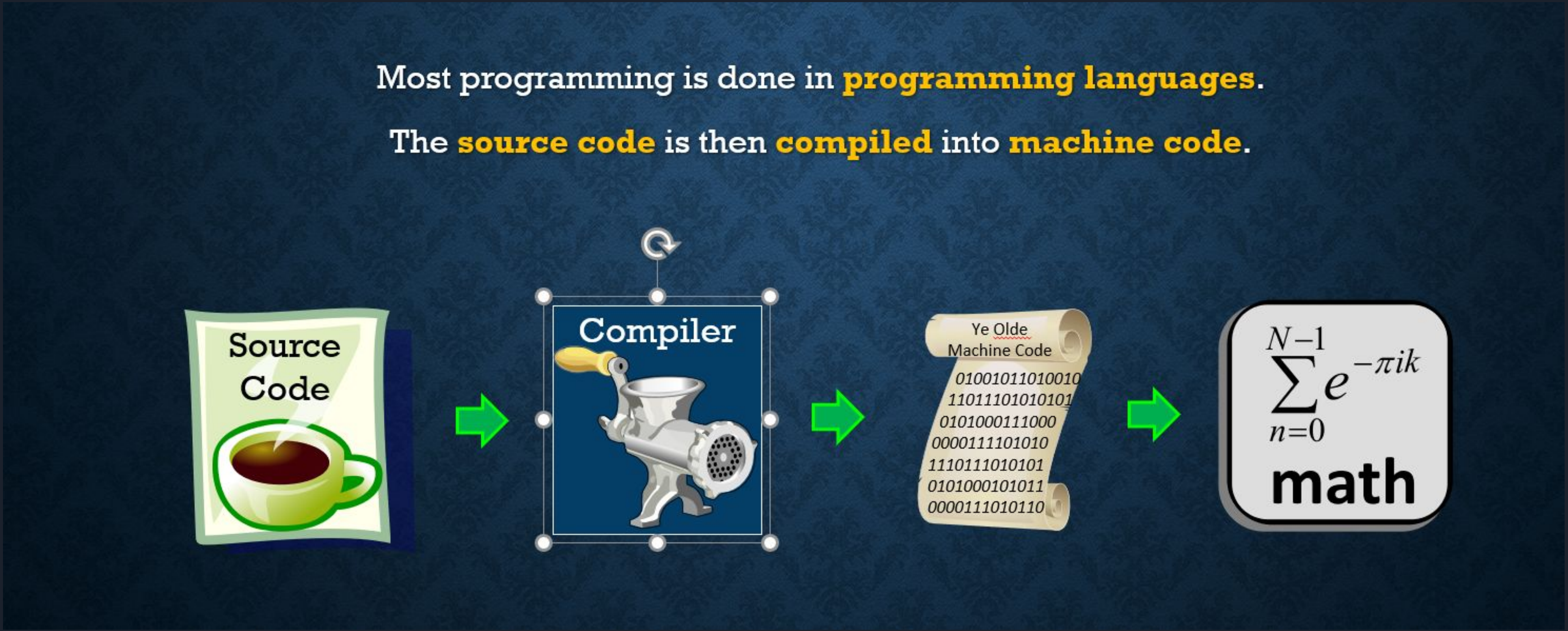
Exam 1 Review

02/08/2022

Module 0



Most programming is done in **programming languages**.

The **source code** is then **compiled** into **machine code**.

Source Code
Compiler
Ye Olde Machine Code
01001011010010
11011101010101
0101000111000
0000111101010
1110111010101
0101000101011
0000111010110
$\sum_{n=0}^{N-1} e^{-\pi i k}$
math

Questions!

1) Describe the compilation process in Java

2) What do standalone statements always end with?

   a. A period

   b. A semicolon

   c. A colon

   d. A smiley face

Answers!

1) Describe the compilation process in Java

   Source code -> compiler -> machine code

2) What do standalone statements always end with?

   a. A period

   b. A semicolon

   c. A colon

   d. A smiley face

Questions!

True/False:

Programs written in the high-level language of a given type can be directly executed by CPU.

High-level language programs must be translated into machine language before they can be executed.

What does the Java compiler do?

A. Translate byte code into machine code

B. Translate source code into machine code

C. Translate machine code to source code

D. Translate source code into byte code

Answers!

True/False:

Programs written in the high-level language of a given type can be directly executed by CPU. False

High-level language programs must be translated into machine language before they can be executed. True

What does the Java compiler do?

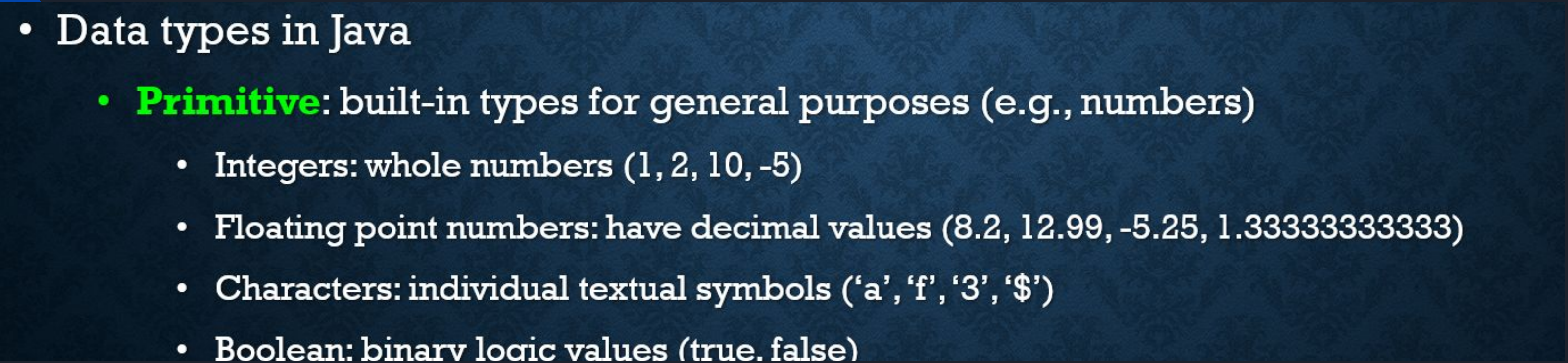
A. Translate byte code into machine code

[✓ B. Translate source code into machine code]

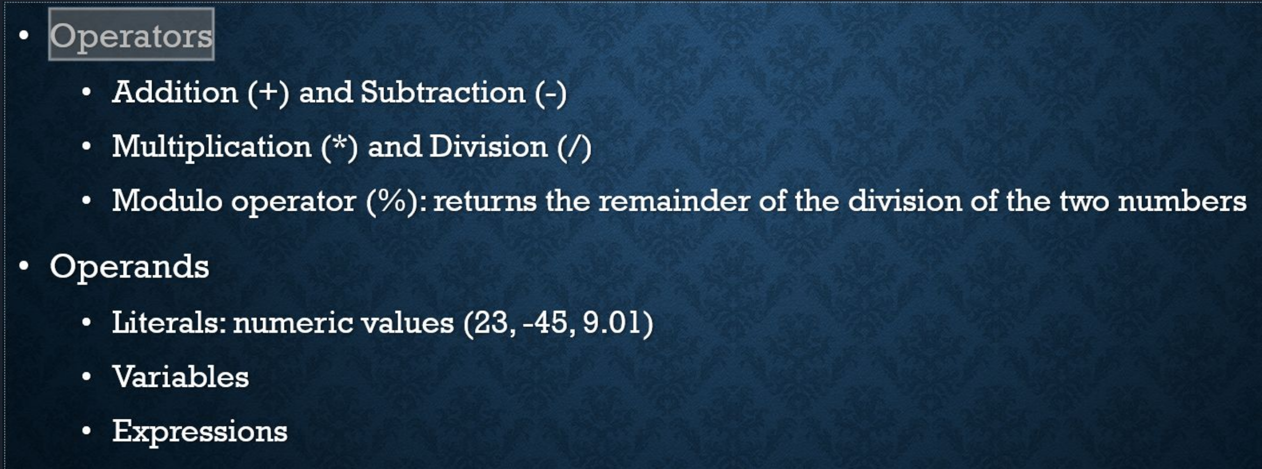
C. Translate machine code to source code

D. Translate source code into byte code

Module 1

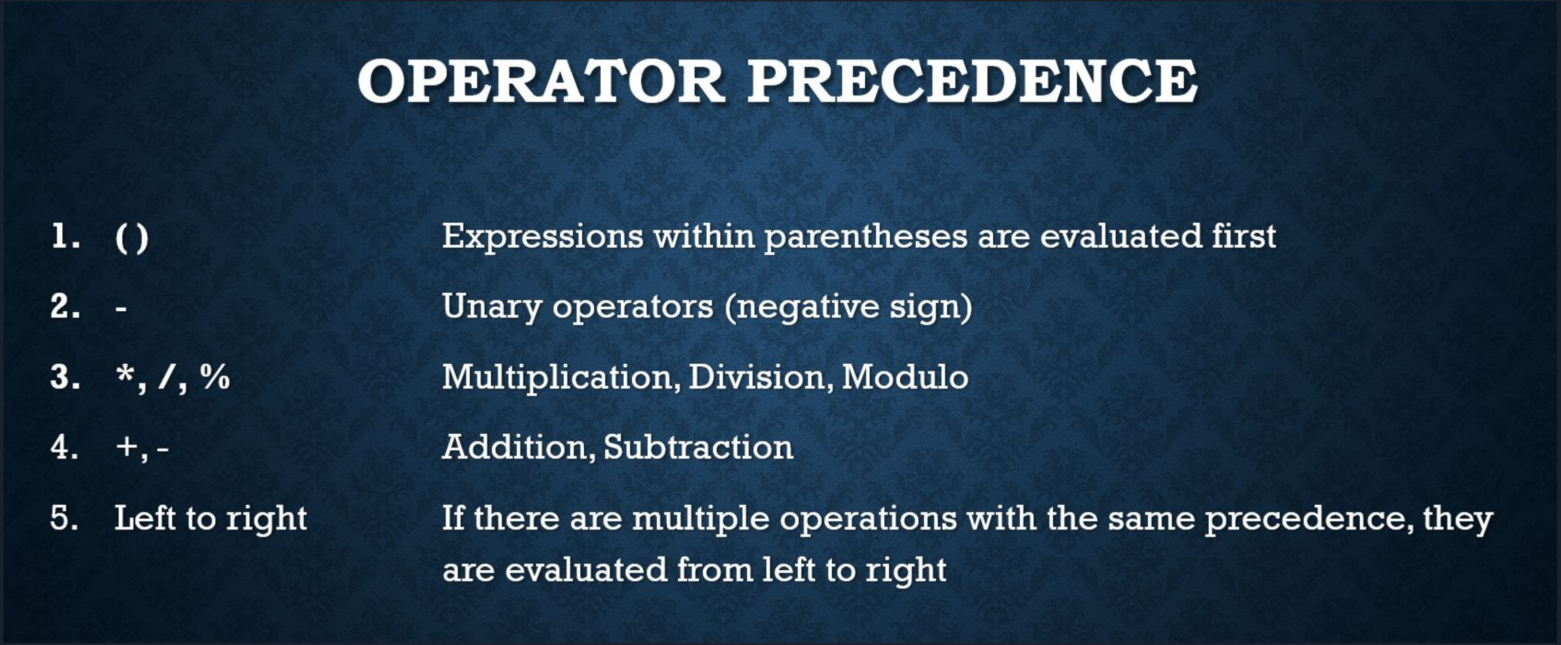


- Data types in Java
  - **Primitive**: built-in types for general purposes (e.g., numbers)
    - Integers: whole numbers (1, 2, 10, -5)
    - Floating point numbers: have decimal values (8.2, 12.99, -5.25, 1.33333333333)
    - Characters: individual textual symbols (‘a’, ‘f’, ‘3’, ‘$’)
    - Boolean: binary logic values (true, false)



- Operators
  - Addition (+) and Subtraction (-)
  - Multiplication (*) and Division (/)
  - Modulo operator (%): returns the remainder of the division of the two numbers
- Operands
  - Literals: numeric values (23, -45, 9.01)
  - Variables
  - Expressions

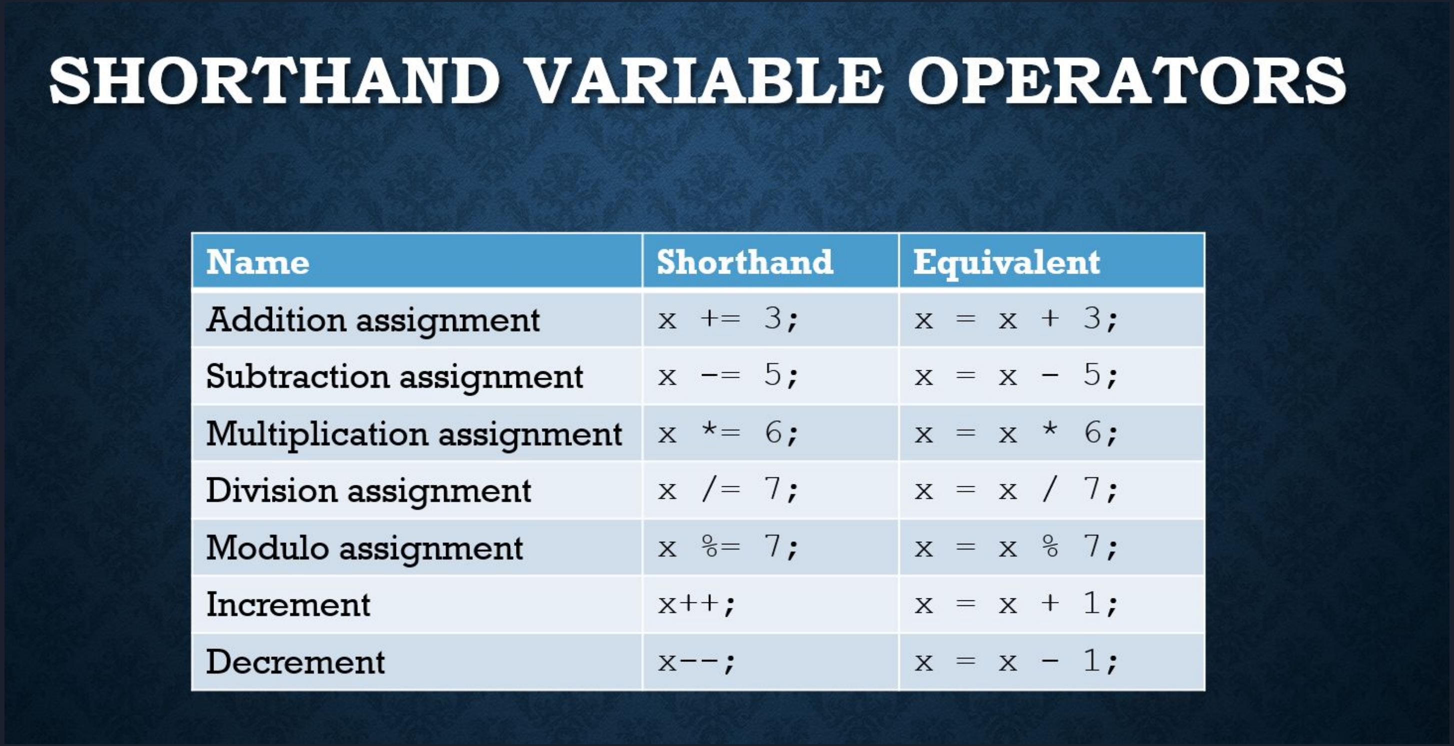
More on operators...



OPERATOR PRECEDENCE

1.	( )	Expressions within parentheses are evaluated first
2.	-	Unary operators (negative sign)
3.	*, /, %	Multiplication, Division, Modulo
4.	+, -	Addition, Subtraction
5.	Left to right	If there are multiple operations with the same precedence, they are evaluated from left to right

What if I don’t want to type as much?


SHORTHAND VARIABLE OPERATORS

Name	Shorthand	Equivalent
Addition assignment	`x += 3;`	`x = x + 3;`
Subtraction assignment	`x -= 5;`	`x = x - 5;`
Multiplication assignment	`x *= 6;`	`x = x * 6;`
Division assignment	`x /= 7;`	`x = x / 7;`
Modulo assignment	`x %= 7;`	`x = x % 7;`
Increment	`x++;`	`x = x + 1;`
Decrement	`x--;`	`x = x - 1;`

Question!

What is the output of these snippets?

1) System.out.println(3/2);
2) System.out.println(3.0/2.0);
3) System.out.println(3.0/2);

Answer!

What is the output of these snippets?

1) System.out.println(3/2);

   1

2) System.out.println(3.0/2.0);

   1.5

3) System.out.println(3.0/2);

   1.5

Question!

What is the output of the snippet?

```
int myNum = 17;
int remainder = myNum % 5;
System.out.print(remainder);
```

Answer!

What is the output of the snippet?

```
int myNum = 17;
int remainder = myNum % 5;
System.out.print(remainder);
```

2
This is because 17/5 = 3 so we remove (5*3) from 17, leaving us with 2!

Question!

What is wrong with this code?

```
int x = 3;
int y = 2;
int y = 3 + x
```

Answer!

What is wrong with this code?

```
int x = 3;
int y = 2;
int y = 3 + x
```

The variable y is declared twice, on line 3, it should just be:

```
y = 3 + x;
```

Question!

1. What are the valid statements? (Assume all the variables are declared e.g. int num = 7;)

A. num = num + 1;

B. num + 3 = 12;

C. num + 2 = num;

D. num = 11 - 8;

E. 8 = num;

Answer!

1. What are the valid statements? (Assume all the variables are declared e.g. int num = 7;)

[✓ A. num = num + 1;]

B. num + 3 = 12;

C. num + 2 = num;

[✓ D. num = 11 - 8;]

E. 8 = num;

Question!

What does the following code output?

```
int x = 10;
int y = 10;
x -= 3;
y++;

System.out.print(x);
System.out.print(y);
```

Answer!

What does the following code output?

```
int x = 10;
int y = 10;
x -= 3;
y++;

System.out.print(x);
System.out.print(y);
```

```
711
```

Question!

Which code snippets have an error?

A. int days; int weeks; days = 7 * weeks;

B. int days; int weeks; int days = 7 * weeks;

C. int days; int weeks = 4; int days = 7* weeks;

D. int weeks = 4; int days = weeks * 7;

E. int days; int weeks = 12; days = 7 * weeks;

F. int days, int weeks = 10; days = 7 * weeks;

G. int days, weeks = 3; days = weeks * 7;

Answer!

Which code snippets have an error?

A. int days; int weeks; days = 7 * weeks;

B. int days; int weeks; int days = 7 * weeks;

C. int days; int weeks = 4; int days = 7* weeks;

D. int weeks = 4; int days = weeks * 7;

E. int days; int weeks = 12; days = 7 * weeks;

F. int days, int weeks = 10; days = 7 * weeks;

G. int days, weeks = 3; days = weeks * 7;

Question

What would be the resulting value of x after this code has executed?

```
int x = 2;
x = x * x + x - x/x + 3;
x = x % 3;
```

A. 0
B. 1
C. 2
D. 3
E. 4

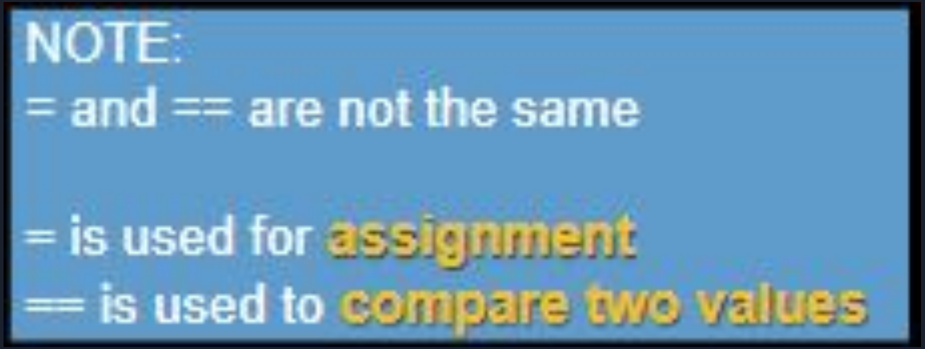
Answer

What would be the resulting value of x after this code has executed?

```
int x = 2;
x = x * x + x - x/x + 3;
x = x % 3;
```

A. 0
B. 1
C. 2
D. 3
E. 4

Module 2

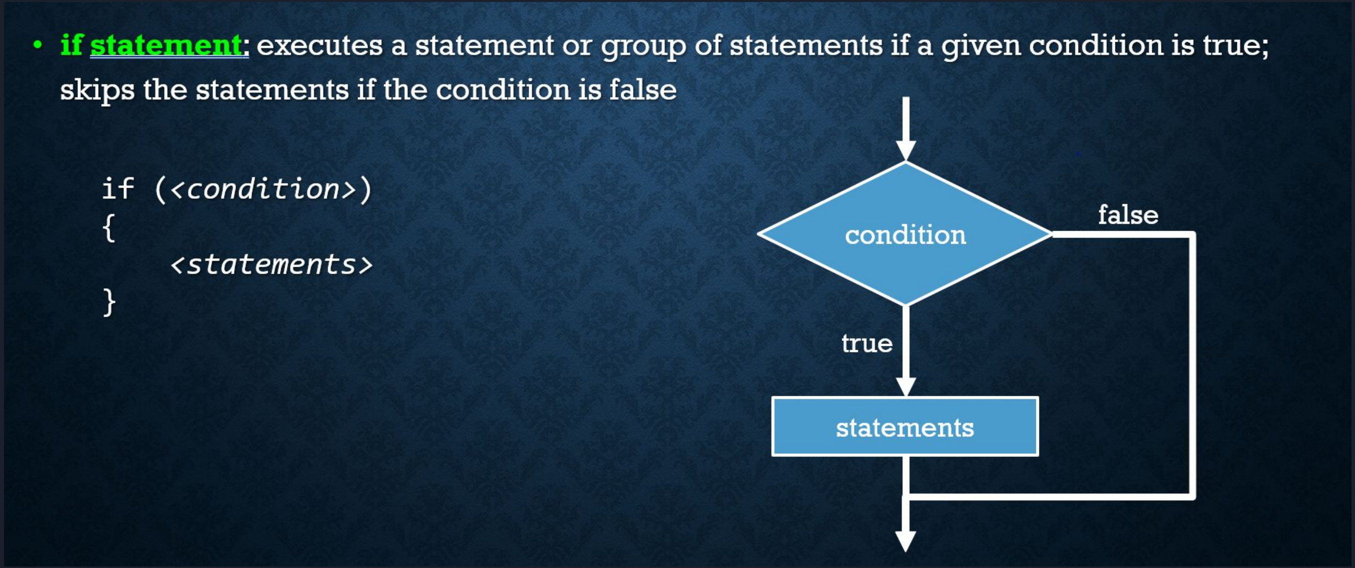


NOTE:

= and == are not the same

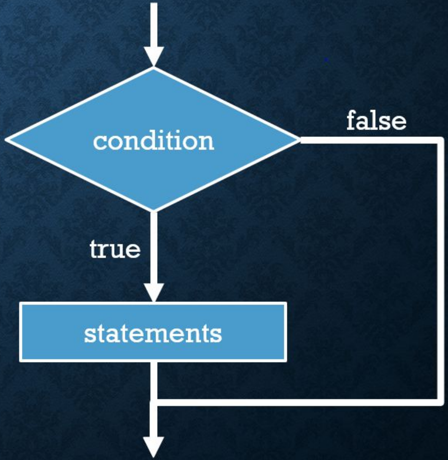
= is used for assignment

== is used to compare two values

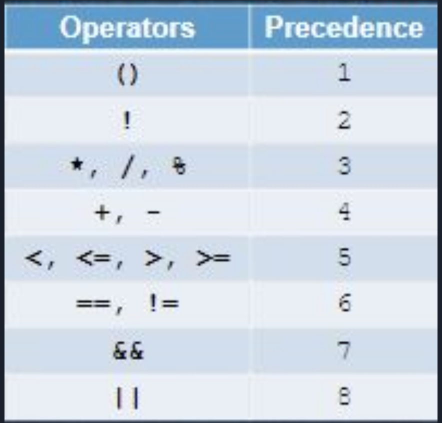


- **if statement**: executes a statement or group of statements if a given condition is true; skips the statements if the condition is false

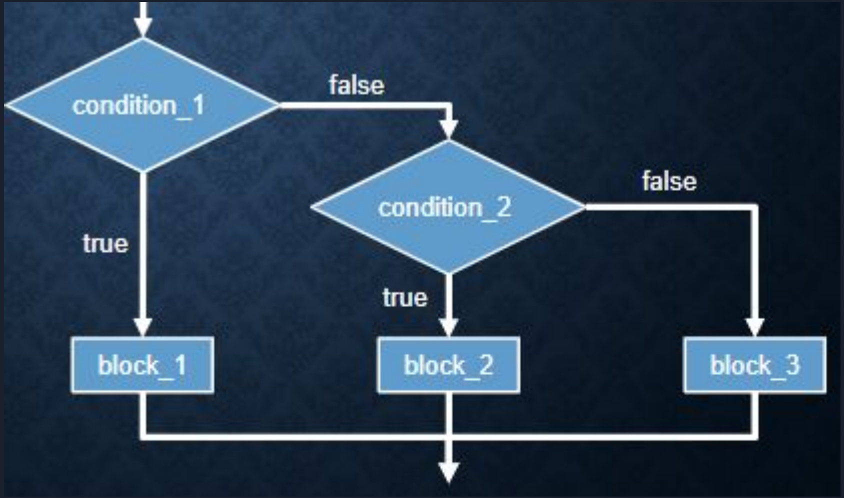
```
if (<condition>)
{
    <statements>
}
```

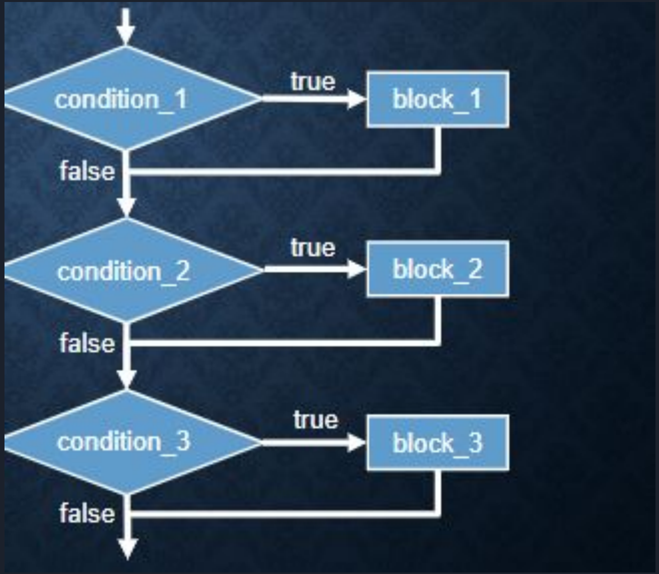


Precedence again??
Plus fancy conditionals!

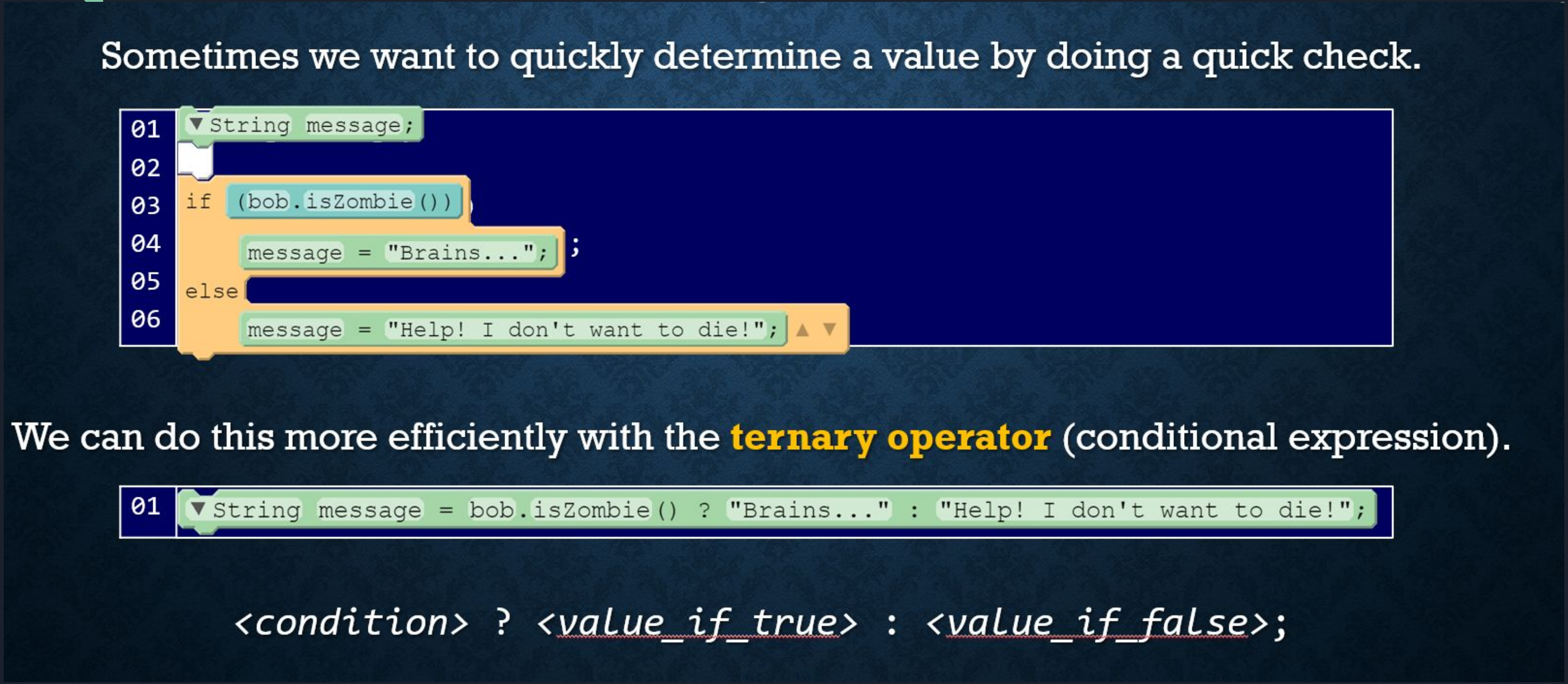


Operators	Precedence
()	1
!	2
*, /, %	3
+, -	4
<, <=, >, >=	5
==, !=	6
&&	7
||	8





Ternary Operator



Sometimes we want to quickly determine a value by doing a quick check.

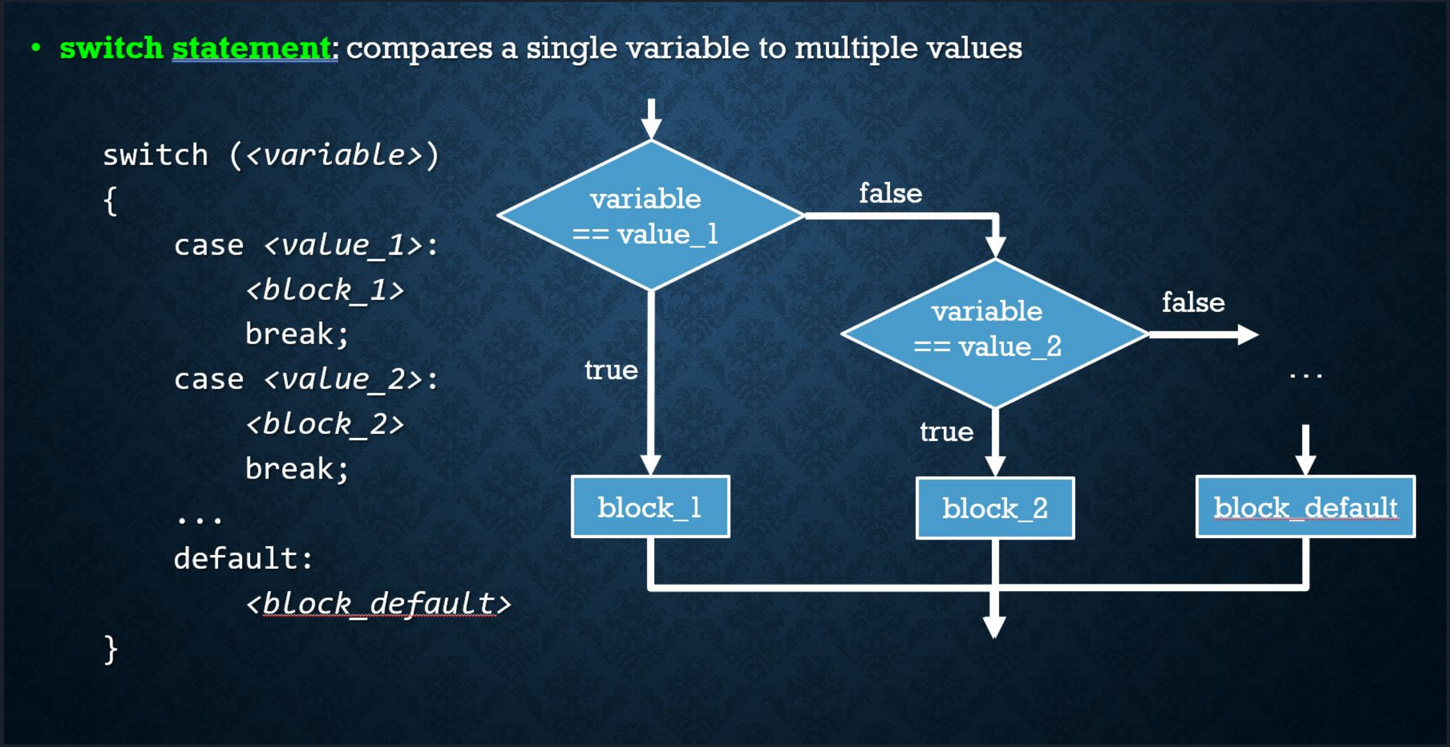
```
▼String message;

if (bob.isZombie())
    message = "Brains...";  ;
else
    message = "Help! I don't want to die!"; ▲ ▼
```

We can do this more efficiently with the **ternary operator** (conditional expression).

```
▼String message = bob.isZombie() ? "Brains..." : "Help! I don't want to die!";
```

```
<condition> ? <value_if_true> : <value_if_false>;
```

Switch Statement



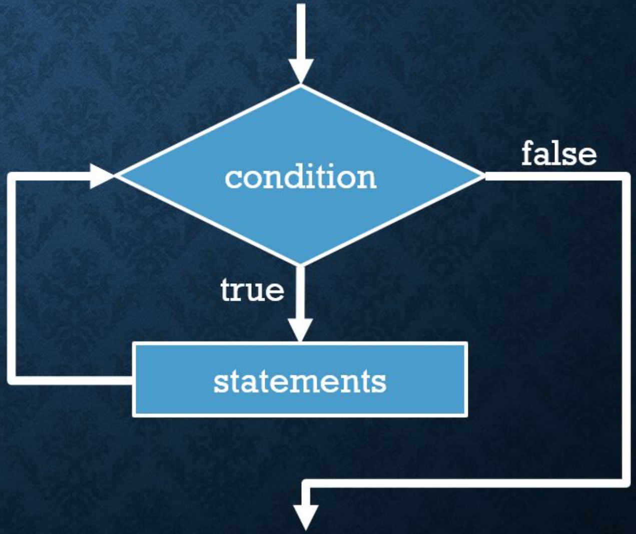
- **switch statement**: compares a single variable to multiple values

```
switch (<variable>)
{
     case <value_1>:
          <block_1>
          break;
     case <value_2>:
          <block_2>
          break;
     ...
     default:
          <block_default>
}
```

While Loops

- **while loops**: executes a statement or group of statements as long as a given condition is true; exits the loop once the condition is false

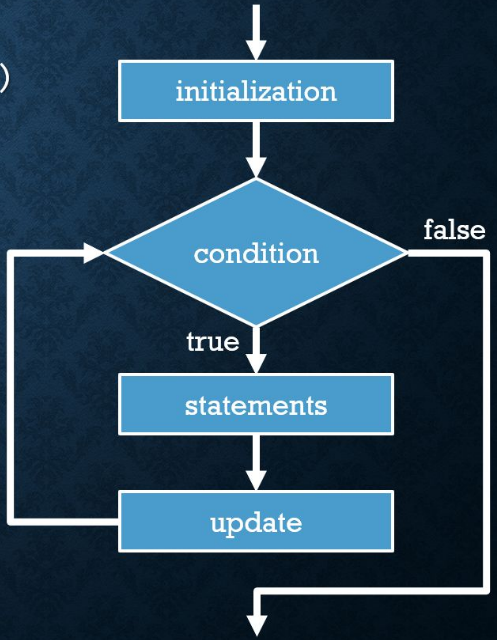
```
while (<condition>)
{
     <statements>
}
```



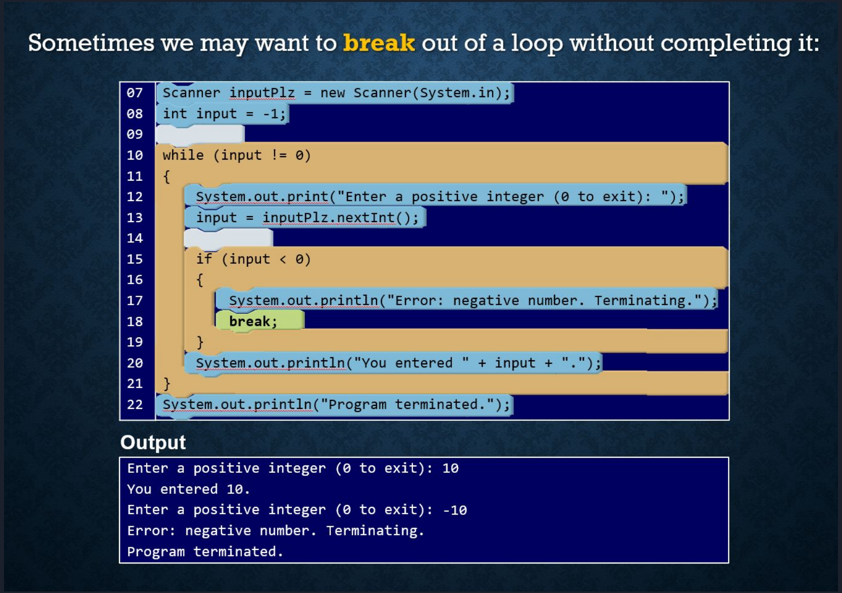
For Loops

- **for loops**: executes a statement or group of statements a specified number of times based on the given initialization, condition, and update expressions

```
for (<initialization>; <condition>; <update>)
{
    <statements>
}
```



Break and Continue

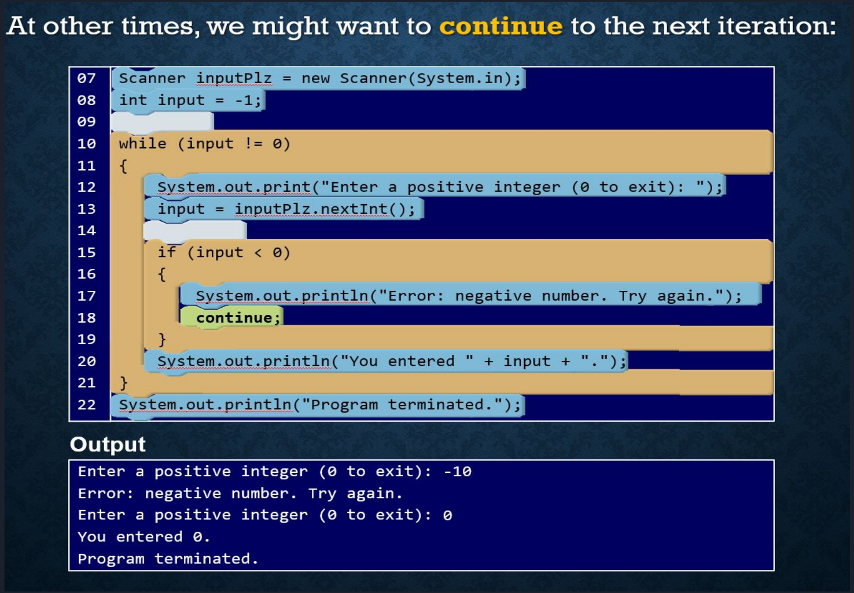


Sometimes we may want to **break** out of a loop without completing it:

```java
Scanner inputPlz = new Scanner(System.in);
int input = -1;

while (input != 0)
{
    System.out.print("Enter a positive integer (0 to exit): ");
    input = inputPlz.nextInt();

    if (input < 0)
    {
        System.out.println("Error: negative number. Terminating.");
        break;
    }
    System.out.println("You entered " + input + ".");
}
System.out.println("Program terminated.");
```

Output

```
Enter a positive integer (0 to exit): 10
You entered 10.
Enter a positive integer (0 to exit): -10
Error: negative number. Terminating.
Program terminated.
```



At other times, we might want to **continue** to the next iteration:

```java
Scanner inputPlz = new Scanner(System.in);
int input = -1;

while (input != 0)
{
    System.out.print("Enter a positive integer (0 to exit): ");
    input = inputPlz.nextInt();

    if (input < 0)
    {
        System.out.println("Error: negative number. Try again.");
        continue;
    }
    System.out.println("You entered " + input + ".");
}
System.out.println("Program terminated.");
```

Output

```
Enter a positive integer (0 to exit): -10
Error: negative number. Try again.
Enter a positive integer (0 to exit): 0
You entered 0.
Program terminated.
```

Question!

What is the output of the following program? (Choose Error if the compiler will generate an error message.)

```
boolean gameOver = false;
if (gameOver = true)
   System.out.println("game is over");
else
   System.out.println("You can continue playing");
```

A. game is over
B. You can continue playing
C. Error

Answer!

What is the output of the following program? (Choose Error if the compiler will generate an error message.)

```
boolean gameOver = false;
if (gameOver = true)
   System.out.println("game is over");
else
   System.out.println("You can continue playing");
```

A. game is over
B. You can continue playing
C. Error

Question!

What does the following code output?

```
char letter = ‘c’;

if (letter > ‘a’ && letter < ‘d’ && letter == ‘A’ || letter == 99)
{
    System.out.println(“true”);
}
else
{
    System.out.println(“false”);
}
```

Answer!

What does the following code output?

```
char letter = ‘c’;

if (letter > ‘a’ && letter < ‘d’ && letter == ‘A’ || letter == 99)
{
    System.out.println(“true”);
}
else
{
    System.out.println(“false”);
}
```

true

Question!

What does the following code output?

```
boolean want_cookie = true;
boolean answer = want_cookie ? 3 > 4 : 1 < 2;
System.out.println(answer);
```

Answer!

What does the following code output?

```
boolean want_cookie = true;
boolean answer = want_cookie ? 3 > 4 : 1 < 2;
System.out.println(answer);
```

False
The ternary evaluates to the 3>4 which is a false statement and this is what is returned

Question!

What is wrong with this segment of code?

```
else if (int > 10)
{
     System.out.print(“The value is greater than 10”);
}
else
{
    System.out.print(“You have a number less than
ten”);
}
```

Answer!

What is wrong with this segment of code?

```
else if (int > 10)
{
     System.out.print(“The value is greater than 10”);
}
else
{
    System.out.print(“You have a number less than
ten”);
}
```

If must be present before an else if statement

Question!

What is the output of the following program?

```
int num = 12;
while (num > 0)
{
   num = num - 6;
   System.out.print(num + " ");
}
```

A. 12 6 0
B. 12 6
C. 6 0
D. 6
E. 0

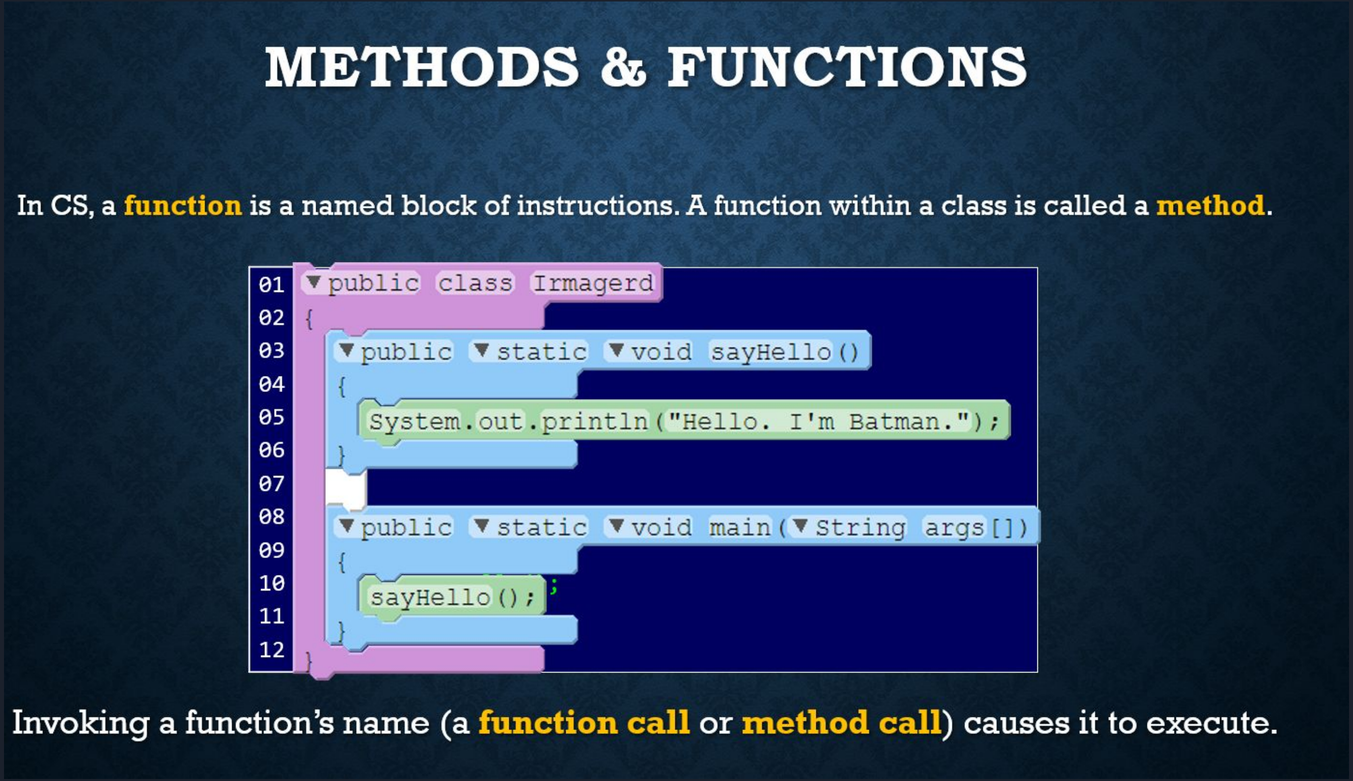
Answer!

4. What is the output of the following program?

```
int num = 12;
while (num > 0)
{
   num = num - 6;
   System.out.print(num + " ");
}
```

A. 12 6 0
B. 12 6
C. 6 0
D. 6
E. 0

Module 3



METHODS & FUNCTIONS

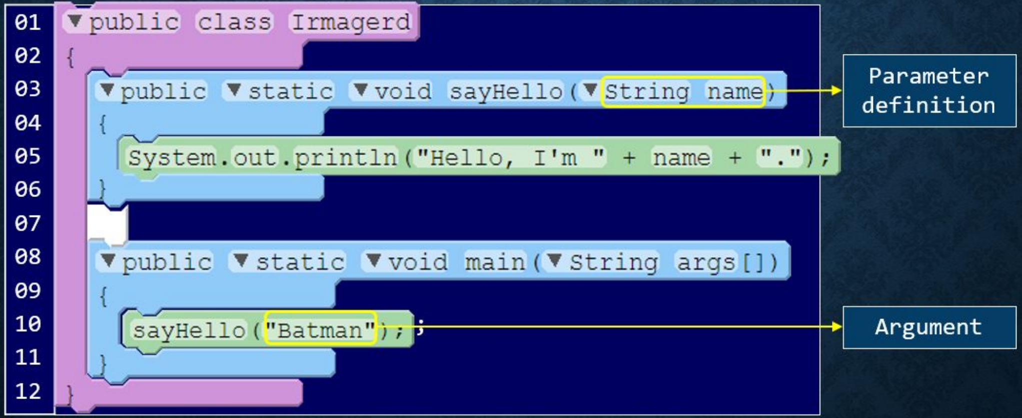
In CS, a **function** is a named block of instructions. A function within a class is called a **method**.

```
public class Irmagerd
{
    public static void sayHello()
    {
        System.out.println("Hello. I'm Batman.");
    }

    public static void main(String args[])
    {
        sayHello();
    }
}
```

Invoking a function's name (a **function call** or **method call**) causes it to execute.

PARAMETERS & ARGUMENTS

We can establish that a function requires **parameters**. Values passed in are **arguments**.



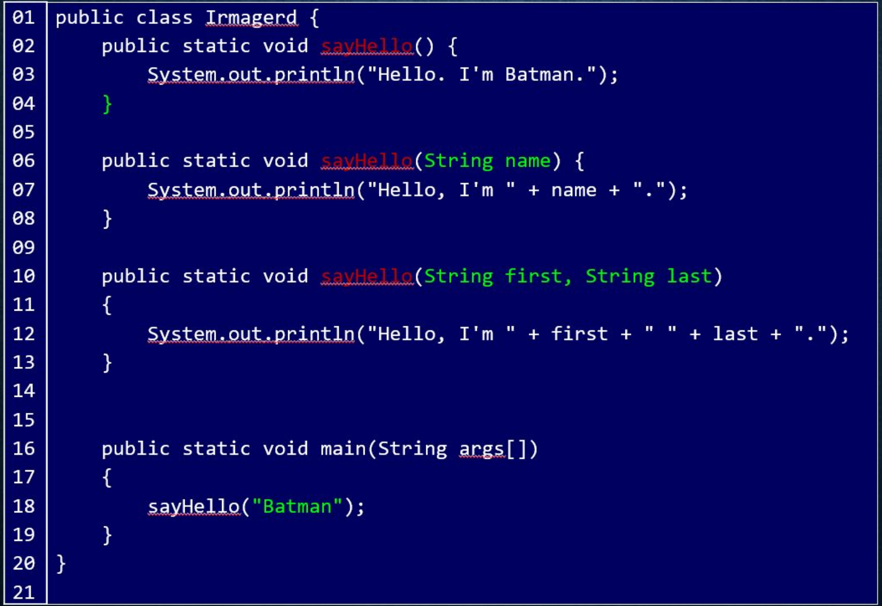
```
▼public class Irmagerd
{
    ▼public ▼static ▼void sayHello(▼String name)
    {
        System.out.println("Hello, I'm " + name + ".");
    }

    ▼public ▼static ▼void main(▼String args[])
    {
        sayHello("Batman");
    }
}
```

OUTPUT:

```
Hello, I'm Batman.
```

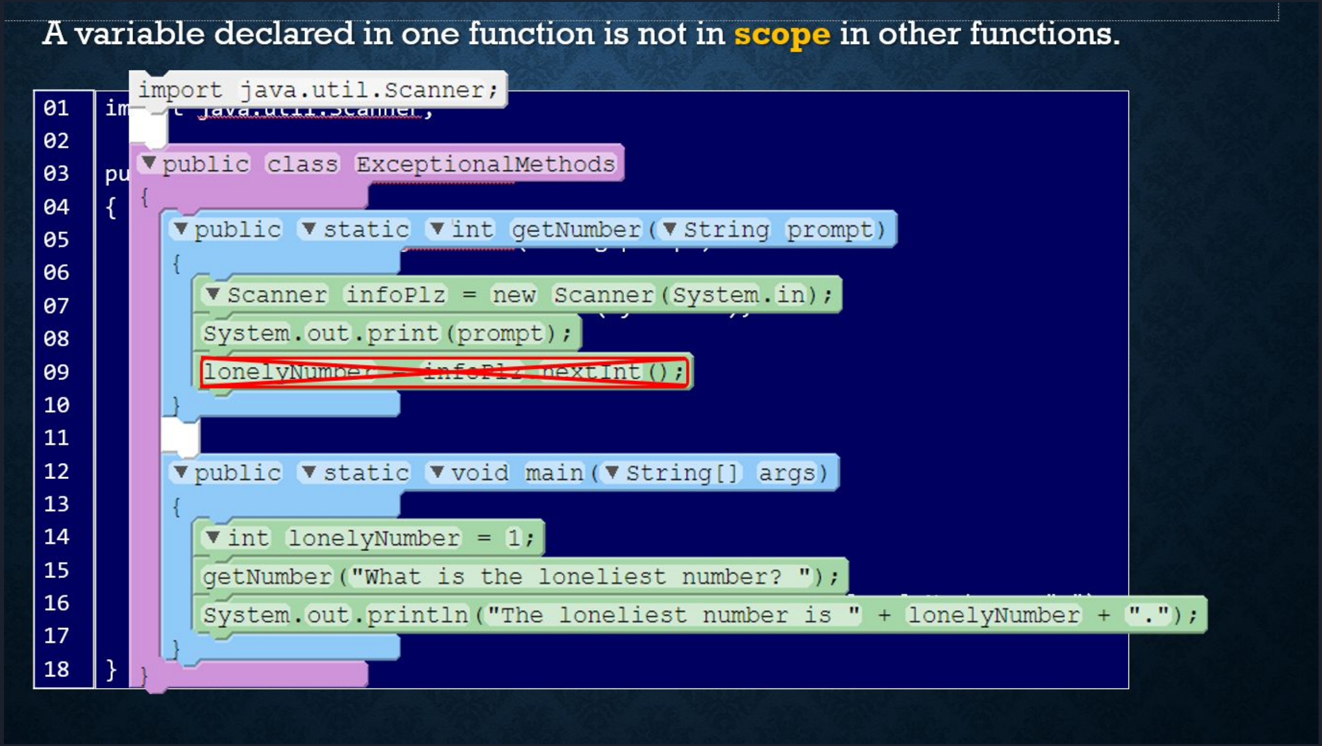
METHOD OVERLOADING

Method Overloading allows a class to have more than one method having the same name with different parameter list



```java
public class Irmagerd {
    public static void sayHello() {
        System.out.println("Hello. I'm Batman.");
    }

    public static void sayHello(String name) {
        System.out.println("Hello, I'm " + name + ".");
    }

    public static void sayHello(String first, String last)
    {
        System.out.println("Hello, I'm " + first + " " + last + ".");
    }


    public static void main(String args[])
    {
        sayHello("Batman");
    }
}

```

Scope for Methods



A variable declared in one function is not in **scope** in other functions.

```
import java.util.Scanner;

public class ExceptionalMethods
{
    public static int getNumber(String prompt)
    {
        Scanner infoPlz = new Scanner(System.in);
        System.out.print(prompt);
        ~~lonelyNumber = infoPlz.nextInt();~~
    }

    public static void main(String[] args)
    {
        int lonelyNumber = 1;
        getNumber("What is the loneliest number? ");
        System.out.println("The loneliest number is " + lonelyNumber + ".");
    }
}
```

Question!

What is the output of the following?

```
public static void main(String [] args)
{
   int x = 1;
   int y = 2;
   int z = 3;
   z = sum(y, z);
   System.out.println(z);
}
public static int sum(int x, int y)
{
   x = x + 10;
   int z = x + y;
   return z;
}
```

Answer!

What is the output of the following?

```
public static void main(String [] args)
{
   int x = 1;
   int y = 2;
   int z = 3;
   z = sum(y, z);
   System.out.println(z);
}
public static int sum(int x, int y)
{
   x = x + 10;
   int z = x + y;
   return z;
}
```

Solution: 15

Question!

```
public static void swap(int a, int b) {
      int temp = a;
      a = b;
      b = temp;
   }
   public static void main(String[] args) {
      int a = 4, b = 5;
      swap(a,b);
      System.out.println(a + " " + b);
   }
```

A.4 5
B. 5 4
C.4 4
D.5 5
E. Error

Answer!

```
public static void swap(int a, int b) {
      int temp = a;
      a = b;
      b = temp;
   }
   public static void main(String[] args) {
      int a = 4, b = 5;
      swap(a,b);
      System.out.println(a + " " + b);
   }
```

A.4 5
B. 5 4
C.4 4
D.5 5
E. Error

Question!

Convert decimal number 78 to binary

Convert hex number 127 to binary

Convert octal number 54 to binary

Answer!

Convert decimal number 78 to binary

1001110

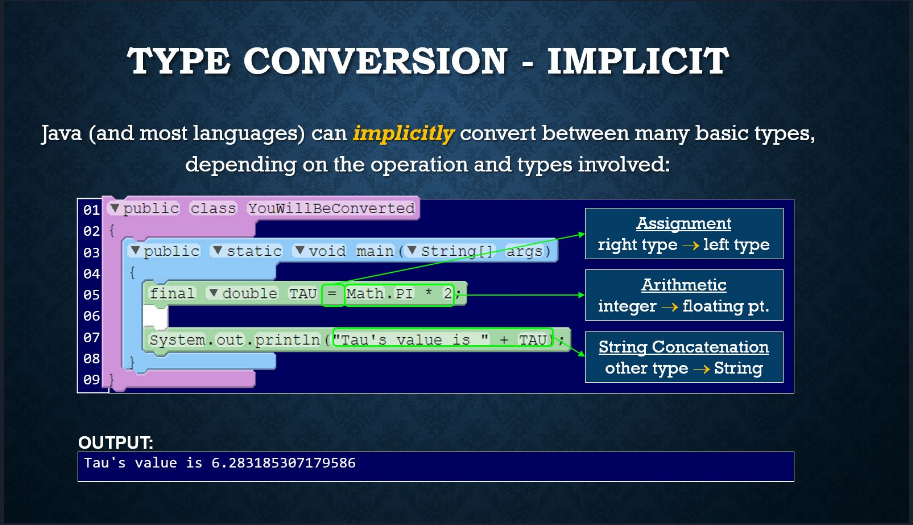
Convert hex number 127 to binary

100100111

Convert octal number 54 to binary

101100

Module 4



TYPE CONVERSION - IMPLICIT

Java (and most languages) can ***implicitly*** convert between many basic types, depending on the operation and types involved:

```
▼public class YouWillBeConverted
{
    ▼public ▼static ▼void main(▼String[] args)
    {
        final ▼double TAU = Math.PI * 2;

        System.out.println("Tau's value is " + TAU);
    }
}
```

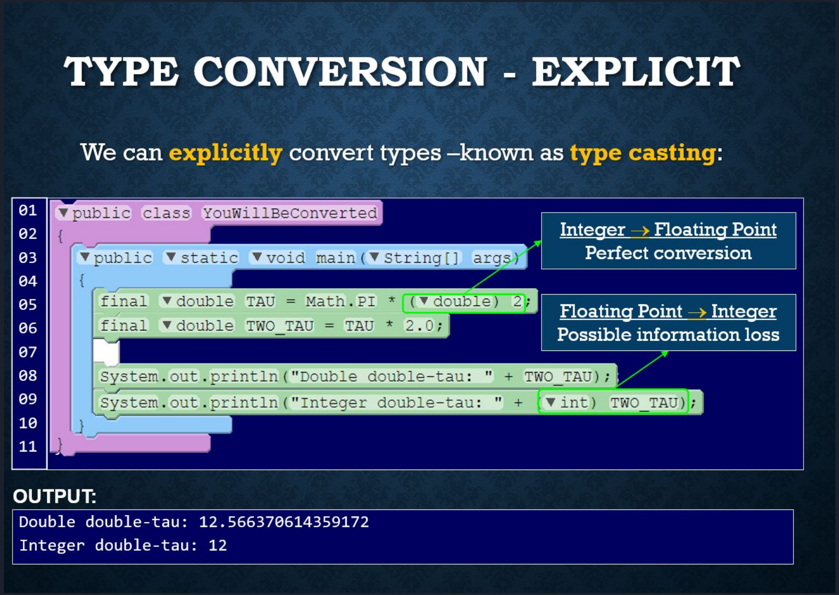
Assignment
right type → left type

Arithmetic
integer → floating pt.

String Concatenation
other type → String

OUTPUT:

```
Tau's value is 6.283185307179586
```



TYPE CONVERSION - EXPLICIT

We can **explicitly** convert types –known as **type casting**:

```
▼public class YouWillBeConverted
{
    ▼public ▼static ▼void main(▼String[] args)
    {
        final ▼double TAU = Math.PI * (▼double) 2;
        final ▼double TWO_TAU = TAU * 2.0;

        System.out.println("Double double-tau: " + TWO_TAU);
        System.out.println("Integer double-tau: " + (▼int) TWO_TAU);
    }
}
```

Integer → Floating Point
Perfect conversion

Floating Point → Integer
Possible information loss

OUTPUT:

```
Double double-tau: 12.566370614359172
Integer double-tau: 12
```

Question!

```
public static void main(String[] args) {
      String a = "hello" + 12 + 5;
      System.out.println(a);
}
```

Question!

```
public static void main(String[] args) {
        String a = "hello" + 12 + 5;
        System.out.println(a);
}
```

Solution: hello125

Question!

What is the result of the following?

```
(float) 27/10

(float) (27/10)
```

Answer!

What is the result of the following?

(float) 27/10 = 2.7

(float) (27/10) = 2.0

Question!

What is the result of the following?

```
String welcome = “Quack... I am a ducky”;
char letter1 = welcome.charAt(4);
char letter2 = welcome.charAt(7);
System.out.print(letter1);
System.out.print(letter2);
```

Answer!

What is the result of the following?

```
String welcome = “Quack... I am a ducky”;
char letter1 = welcome.charAt(4);
char letter2 = welcome.charAt(7);
System.out.print(letter1);
System.out.print(letter2);
```

k.

Question!

Predict program output:

```
public static void main(String[] args){
      int i = 6;
      switch(2%i)
      {
                 case 2:
                  System.out.print("2");
            case 3:
                  System.out.print("3");
                  break;
                 case 4:
                  System.out.print("4");
            case 5:
                 System.out.print("5");
      }
}
```

Answer!

Predict program output:

```
public static void main(String[] args){
      int i = 6;
      switch(2%i)
      {
                case 2:
                 System.out.print("2");
           case 3:
                 System.out.print("3");
                 break;
                case 4:
                 System.out.print("4");
           case 5:
                System.out.print("5");
      }
}
23
```

Coding Question!

Write a method `numMultiplesOfTwo(int low, int high)` whose output is the number of multiples of `2` between `low` and `high` inclusive.

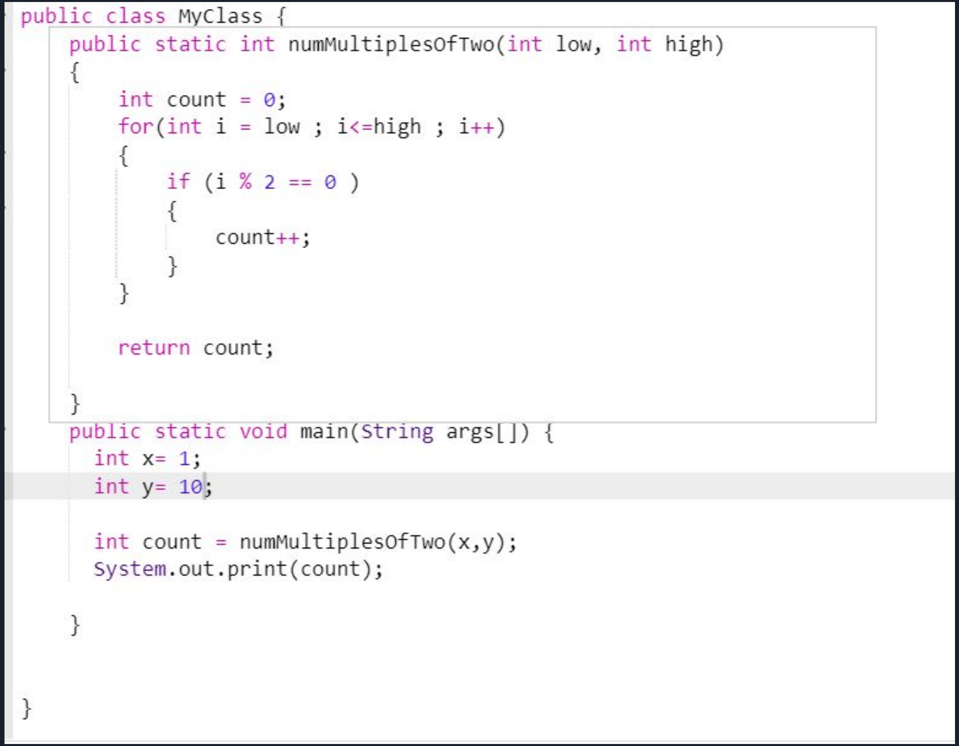
E.g.1.

- input: low = 1, high = 10

- returned value: 5

- Explanation: in the range of [1, 10] both inclusive, the numbers 2, 4, 6, 8, 10 are multiples of 2. There are a total of 5 numbers.

Possible Solution!



```
public class MyClass {
    public static int numMultiplesOfTwo(int low, int high)
    {
        int count = 0;
        for(int i = low ; i<=high ; i++)
        {
            if (i % 2 == 0 )
            {
                count++;
            }
        }

        return count;

    }
    public static void main(String args[]) {
      int x= 1;
      int y= 10;

      int count = numMultiplesOfTwo(x,y);
      System.out.print(count);

    }


}
```

Note: The question said you do not need to write a main method, so only the numMultiplesOfTwo method is required for this question.