****

Section 1 : Output(print out)

Section 2 : Variable

Section 3 : Equation

Section 4 : Input

Section 5 : Tips

Base of Programing

**Chapter 2**

Section 1. Output(print out)

How do we talk with computer by code?

Let’s think about how we talk each other. We make sounds with mouth and tongue. And there are rules in language. We call it grammar and words. This logic works in programing too. We write code to talk. And there are rules in programing languages.

Let’s start with analyzing our first code (the code while testing Eclipse)

01 public class FirstJavaProgram {

02 public static void main(String[]args) {

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

04 }

05 }

Just like reading a book, we start read from the top.

1)bracket

There are many brackets in a program. This brackets are just the same as bracket in math.

For example let’s look at ((5\*3)+2)\*2. There are two pairs of bracket. They are always in pairs and express these numbers are surrounded. Bracket in program express where the instruction starts, and ends. And it makes both computer and people understand code easier.

If some codes are inside a bracket, there will be extra space on the right. You can check line 02. Because it’s in a bracket, there are spaces in front for the code. If you see line 03, it has more spaces than 02. It’s because it’s in 2 brackets. So, the space on the left shows the code with how many brackets.

2)first line

When you followed the step 6 of running Eclipse, there was already some codes.

public class FirstJavaProgram {

}

It shows your program’s name is FirstJavaProgram.

3)Second line(Main method)

We are going to see this code everytime because this code is essential for a program. You are going to make code after this line everytime.

public static void main(String[]args) {

}

You will see these two lines every time. And you will never fix them. All you need to know is this.

All the form of your code will be like this

public class ClassName {

public static void main(String[]args) {

//codes

//codes

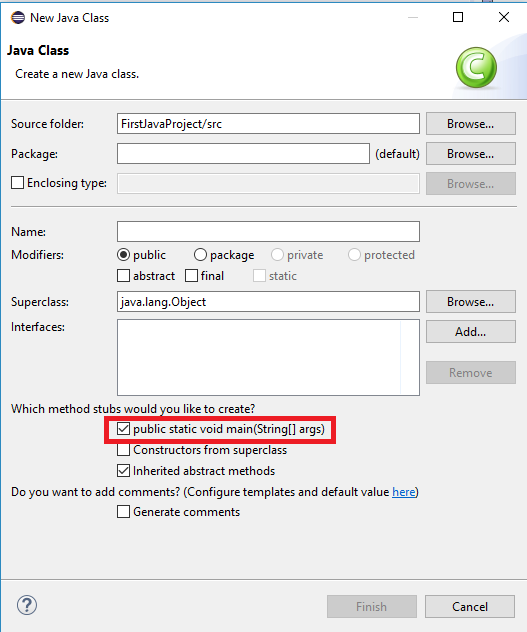
//codes

……

}

}

You will only write code in //codes position.



If you don’t want to write main method, check that box, and the compiler will automatically make that code for you. And you will write code in the main method.

Section 1. output

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

Now we are going to talk about the code you need to write.

This code command to print out things in the bracket.

Standard print statement-1



\*Underlined part is modifiable.

: Print out Sentence and go to the next line.

The importance of semicolon

Computer does not know if the line has been changed. For example,

1+1 = 2

2+2 = 4

When we see these two line, we know they are separated but what computer get is

1+1=22+2=4.

Why?

If you write like this :

1+1=2;

2+2=4;

The computer will recognize it as 2 lines.

**Don’t forget to write ‘ ; ‘ end of each line!**

Let’s make a program with print out

Hello

World!

Example 2-1-1 Output1.java

01 public class Output1{

02 public static void main(String[]args) {

\*03 System.out.println(“Hello”);

\*04 System.out.println(“World!”);

05 }

06 }

Explain

03 : Print “Hello” on the first line.

04 : Print “World!” on the next line.

output

Hello

World!

**All you need to see in this program is the line where there are stars in front of line numbers.**

P1. Let’s make a program that prints “I’m going to be the ace in ACE.”

**o**utput

I’m going to be the ace in ACE.

Standard print statement-2



\*Underlined code is modifiable.

Print the sentence

What’s different from the standard print statement 1?

Example 2-1-2 Output2.java

01 public class Output2 {

02 public static void main(String[]args) {

\*03 System.out.print(“Hello”);

\*04 System.out.print(“World!”);

05 }

06 }

Explain

03 : print “Hello” on the first line

04 : print “World!” on the first line after “Hello

Output

HelloWorld!

It will not change to next line, it will keep printing in the same line.

P2. Let’s make a program that prints “Java is very easy.” using 4 System.out.print().

output

Java is very easy.

If you want to write in one line, you don’t have to write 2 lines of codes. There is a more convenient way.

Standard print statement -3



\*Underlined code is modifiable

You can print different things by using operator +

You can print more than 2 things

Types of printing

Sentence : cover the sentence with double quotation marks. Ex) “Harry Potter”

Numbers : just write the numbers. Ex) 324

Arithmetic equation : cover with bracket. Ex) (1+2).

The four fundamental arithmetic equations in code are + - \* /

There is an example of many different way of printing

Example 2-1-3 Output3.java

01 public class Output3 {

02 public static void main(String[]args) {

\*03 System.out.println(“Hello” + “World!”);

\*04 System.out.println(“I am ” + 15 + “ years old”);

\* 05 System.out.println(“1+1 = “ + (1+1));

06 }

07 }

Explain

03 : Print “Hello” and print “World!” And go to next line(because it’s println)

04 : Print “I am” and print 15 and print “years old” and go to next line

05 : print “1+1 =” and print result of 1+1

output

HelloWorld!

I am 15 years old

1+1=2

There are many other ways to print, but let’s practice with those skills.

P3. Make a program that prints “I can eat 5 hamburgers.” by using 4 + operators

output

I can eat 5 hamburgers.

Questions

A. Guess how the output would be.

practice A-1-1

public class A\_1\_1 {

public static void main(String[]args) {

System.out.print(“Monday”);

System.out.println(“Sunday”);

System.out.print(“Friday”);

}

}

practice A-1-2

public class A\_1\_2 {

public static void main(String[]args) {

System.out.println(“I want it” + “that way”);

System.out.println(“R” + “o” + “l” + “l” + “i” + “n” +”g” );

System.out.println(“10 times 400 is ” + (400\*10));

}

}

B. Find the errors

practice B-1-1

public class A\_1\_3 {

public static void main(String[]args) {

System.out.println(“help” “me”);

System.out.println(8 divided by 2 equal + (8/2) );

System.out.println(“There might be no error”)

}

}

C. make a program

1-1.Output of your program will be :

Output

I don’t want a lot for Christmas

There is just one thing I need

1-2. Make a program that prints out the answer of the equations.

Q1 : 4500 \* 400 / 2 + 7132

Q2 : (432 + 35123) \* (5122 + 9432)

The format of the output is going to be each equation and answer.

Output

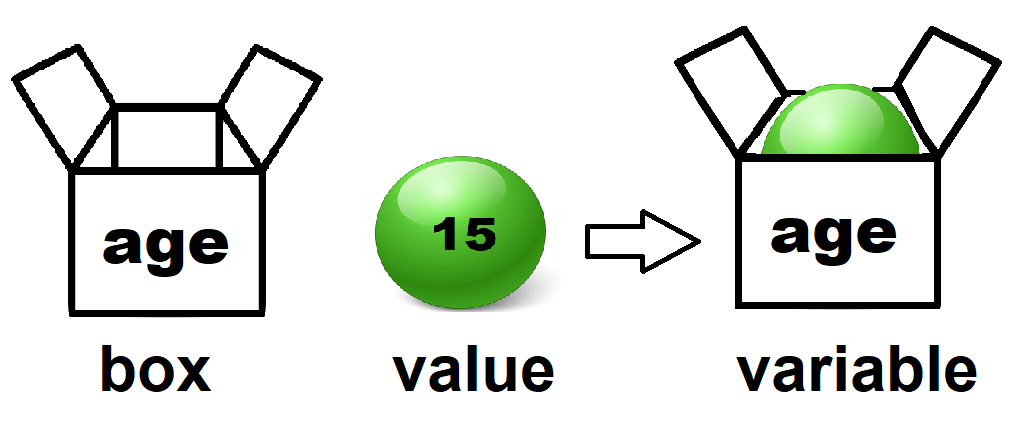
Q1 = Answer

Q2 = Answer

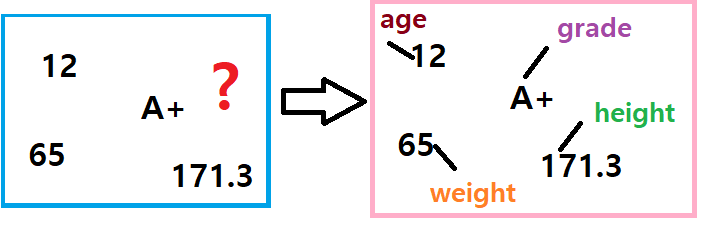
Section 2. Variable.

Variable is not a familiar word. Variable means **a number that can be changed**. For example, Pi(3.14) never changes. So, Pi is not a variable. But your age will be changed next year, so we can call your age is a variable.

In programing, we put a value in a variable box. And we call it a variable.



Let’s say there is value 15. It can be age or years or the number of a family. To make sure what it is, the box has a name.

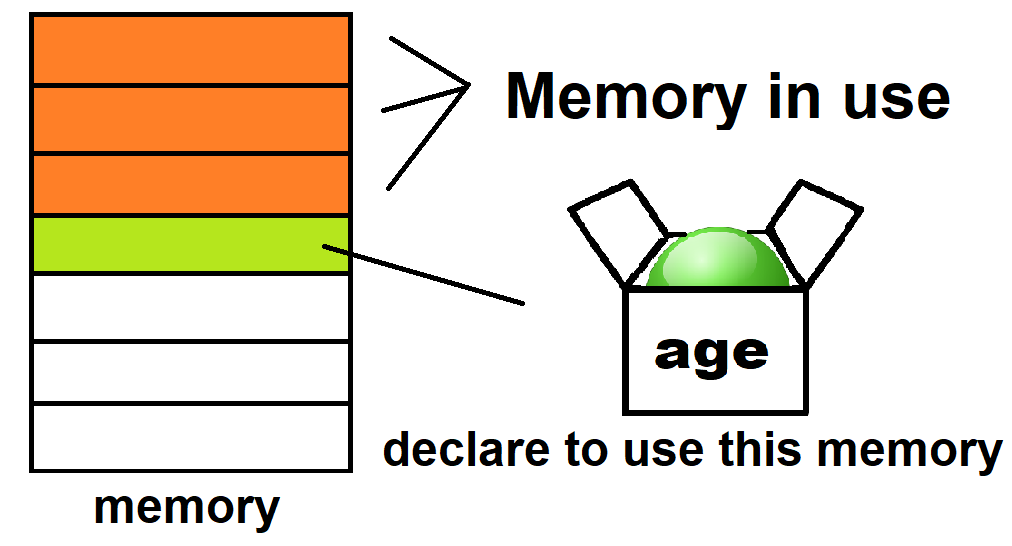


There are going to be tones of data. It would be very hard to recognize the meaning of each data. So we put them in each variable.

There are steps to use variable in a program.

1) Declare

Do you know how big is your hardware memory of your computer? 128GB? Or 1TB? 2TB? We are living in the world with TB memory. We know if we download something, they need memory for them. For example: a mp3 file is about 5MB, a movie is about 2~10GB. When you download something, you must click the file you want to download from internet. And then computer will start downloading. In this process, you are getting permission from the computer to use some memories to store your file. This is just same as declare step. If you want to make a variable, you have to tell computer to **get some memories** **for the variable** so other data won’t make collision with the variable.



What do you think declaring does?

Declare -1



\*Underlined part is modifiable.

Type : the type of variable which depends the size of memory

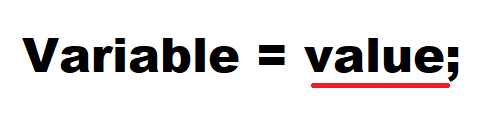
Name : name of the variable(box)

Meaning : Declare a specific type of variable named ‘name’.

2)initialize

After you declare a variable, the variable has no value. The step we give a value to a variable is called initializing.

Initialize-1



\*Underlined part is modifiable.

Meaning : put a value in a variable.

This program make a variable named age, and this variable contains the information of your age.

It will print out how old you are.

Example 2-2-1 Type1.java

01 **public** **class** Type1 {

02 **public** **static** **void** main(String[] args) {

03

\*04 **int** age;

05

\*06 age = 15;

07

\*08 System.***out***.println("I am " + age + " years old");

09 }

10 }

Explain

04 : Declare a variable named age which is an integer type (Declare). There are many different types of a variable. Int is a type of integer.

06 : put value of 15 in to the variable named age.(initialize). “=” does not mean equal in programing. A = B means put the value B to variable A.

08 : Even though you put variable age instead of the value 15, the computer will print 15.

output

I am 15 years old

P4. Make a program that stores a data of height and print it.

output

My height is 180cm.

Ex) A and B are variables

A = 10; : Put 10 to A-> so A’s value equals 10

B = 2 + 3 ; : Put the answer of 2 + 3 to B -> B’s value equals 5

A = B; : Put B’s value to A -> A’s value equals 5

3 essential parts of a variable are

**-Type -Value -Name(box)**

Prior to programing(coding), We need to declare / initialize a variable.

people thought ‘**Why don’t we do these two together?**’ Now, we can declare and initialize together.

Declare - 2



\*Underlined part is modifiable

value : a value witch fits the type.

Meaning : Declare a specific type of variable named ‘name’ and initialize it with value.

For example)

int age = 15; : if you write this code, you can merge line 04 and line 06 in Example 2-1.

To make your code look simple and organized, you can declare multiple variables



\*Underlined part is modifiable

Meaning: Declare a specific type of variables named ‘name1’ and ‘name2’.

For example)

int age, height;

int age, height, weight, grade;

To make your code look simpler and more organized, you can declare and initialize multiple variables



\*Underlined part is modifiable

Meaning: Declare a specific type of variables named ‘name1’ and ‘name2’ and initialize each of them with value1 and value2.

For example)

int age = 15, height = 170;

int age = 15, height = 170, weight = 65, grade = 9;

P5. Make a program.

Conditions:

There will be 3 variables, telling your age, the number of your siblings and how many days you studied java.

Use these variables to print out these lines

Output

I’m 15 years old.

I have 2 siblings.

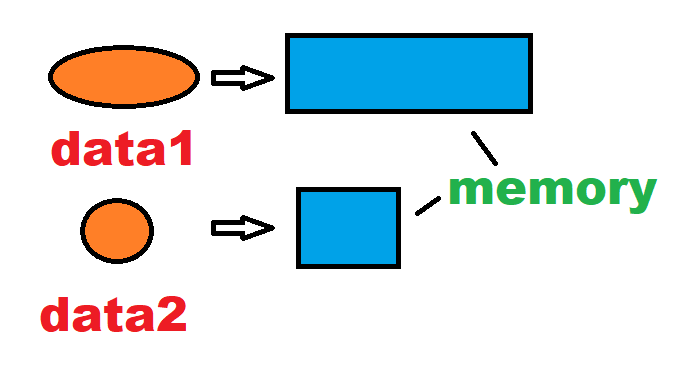
I studied java for 2 days.

3)Type

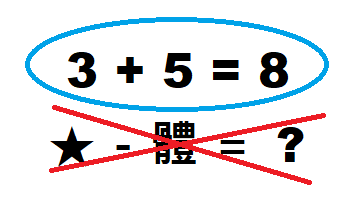
I know all you would think that ‘There was something hard to understand in variable. I know what means value and box. But still don’t get type. What is int?’

Type means specific format of a data. There are a number of types in Java. There are two reasons for using types.

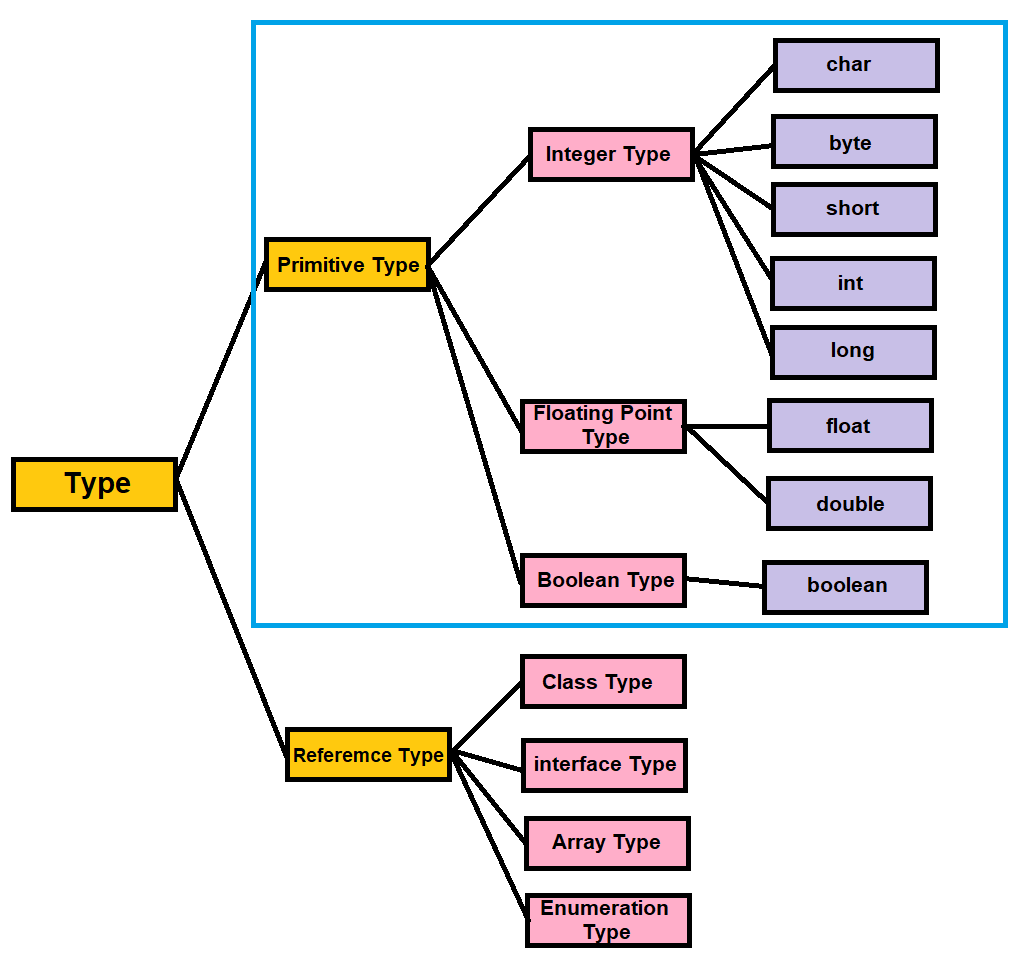
First reason: We need to **take different size of memory** for different type of data.



Second reason: will computer recognize pi + 200 ? Probably not. A word can’t be increased by numbers. Computer needs to know if they can be calculated or not. Only same type of operands can be calculated.



These are types of java



We are only going to learn about the types in the blue box. Primitive type is offered for efficient execution. We can divide it to integer, floating point and Boolean type. They express numbers in different ways.

Chart 1: Types

|  |  |  |  |
| --- | --- | --- | --- |
| Type | Bit | Range | example |
| byte | 8 | -128 ~ 127 | 1 |
| short | 16 | -32,768 ~ 32,767 | 2 |
| int | 32 | -2,147,483,648 ~ 2,147,483,647 | 8 |
| long | 64 | -9,223,372,036,854,775,808 ~ 9,223,372,036,854,775,807 | 15 |
| char | 16 | 0 ~ 65,535 | ‘a’ |
| float | 32 | 1.4E – 45 ~ 3.4028235E38 | 3.14f |
| double | 64 | 4.9E-324 ~ 1.7976931348623157E308 | 3.14d |
| boolean | 8 | 0 ~ 1 | True or 0 |

In the chart, bit means the size of the memory. So, byte and Boolean has the shortest size of memory, long and double has the longest size of memory. It depends the range of the numbers that each type can contain.

[1] integer type (byte, short, int, long, char)

There are 5 types in integer type. 4 of them (byte, short, int, long) are for numbers and ‘char’ is for characters. Integer types only can contain integer numbers which means they **can’t hold fractions or decimal.**

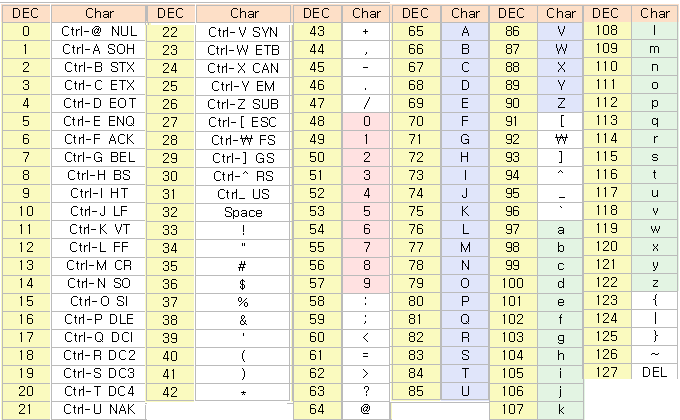
Ex) 1, 300, 843, 23849324

You can save memory by using suitable type.

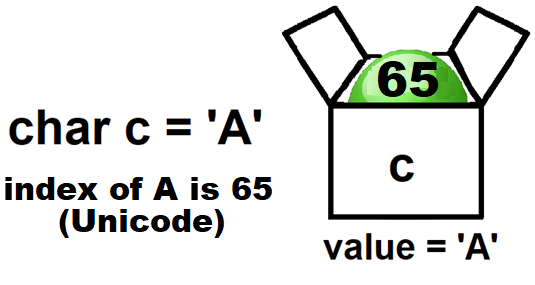


char is a type contains characters. But why is it a integer type? Because each character has their own number. We call it Unicode. Computer memorize many different characters from many other countries with index.

Chart 2 : Unicode



In this chart, Dec means decimal number of each character. So, if I find an index of letter is 65, it will be A in char type. There are 1,048,576 characters in the world. And Unicode has these numbers of characters. So, this chart is just a part of Unicode. To understand this, let’s see this picture.



When you look at the chart, 65 is A. so even you save 65 in char type variable, you will get A.

Let’s make this program and understand what char type is.

Example 2-2-2 CharType.java

**01 public** **class** CharType {

02 **public** **static** **void** main(String[] args) {

03

\*04 **char** c1 = 'A';

\*05 **char** c2 = 65;

06

\*07 System.***out***.println(c1);

\*08 System.***out***.println(c2);

09

10 }

11}

Explain

04 : Declared Variable named ‘c1’ as a char type and initialized to ‘A’

05 : Declared Variable named ‘c1’ as a char type and initialized to 65

07 : print out c1

08 : print out c2

Output

A

A

Now we get the same output.

P6. Make a program. This program has 5 char type variables and each of them are written in number of Askii code. Each of them are ‘A’,’C’,’E’. print “ACE” using these variables.

Output

ACE

There are useful codes to write special characters

Chart 3 : special characters

|  |  |  |
| --- | --- | --- |
| Special characters | Designation method | unicode |
| Backspace | \b | \u0008 |
| Tab | \f | \u0009 |
| Linefeed | \n | \000A |
| Formfeed | \f | \u000C |
| Backslash | \\ | \u005C |
| Single Quote | \’ | \u0027 |
| Double Quote | \” | \u0022 |

Ex)

char c1 = ‘\t’; : c1 = tab

char c2 = ‘\n’; : c2 = linefeed

[2] floating point type(float, double)

Floating point types can hold decimal. The way this type uses memory is totally different from integer type.

There are two advantages which double type has.

1. can hold bigger number.

2. can save more accurate number.

The default type of floating point is double. So, if you write a floating point like 3.14, it will automatically become double type.

Write ‘f’ after the floating number to describe it’s a float type, write ‘d’ or nothing to describe it’s a double type.

This is the program for to understand differences between float and double type

Example 2-2-3 FloatAndDouble.java

01 **public** **class** FloatAndDouble {

02 **public** **static** **void** main(String[] args) {

03

\*04 **float** a = 12345678901234567890.0f;

\*05 **double** b = 12345678901234567890.0;

06

\*07 System.***out***.println("float variable a's value : " + a);

\*08 System.***out***.println("double variable b's value : " + b);

09

\*10 **float** c= 1.0f/3.0f;

\*11 **double** d = 1.0/3.0;

12

\*13 System.***out***.println("float variable c's value : " + c);

\*14 System.***out***.println("double variable d's value : " + d);

15

16 }

17 }

explain

04 : put 12345678901234567890 into float type variable a.(Declare/initialize) .

05 : put 12345678901234567890 into double type variable b(Declare/initialize).

07,08 : each line print out a and b.

10 : put the result of 1/3 into float type variable c

11: put the result of 1/3 into double type variable d

13,14 : each line print out c and d

Output

float variable a's value : 1.2345679E19

double variable b's value : 1.2345678901234567E19

float variable c's value : 0.33333334

double variable d's value : 0.3333333333333333

If you compare the first and second line of the output, what can you infer?

If you compare the third and forth line of the output, what can you infer?

What is 1.2345679E19?

It means 1.2345679 \*

This is the way of expressing huge numbers in Java

float type print until 8th digit of decimal and double type print until 16th digit of decimal.

P7. Make a program that has a variable containing 999999999999999. And print this number without any loss

Output

9.99999999999999E14

How can we print the decimal until we want to see?

Print statement -4



\*underlined parts are modifiable.

You can use formats in this code.

if you write %.5f, you can print until 5th decimal, if you write %.3f, you can print until 3th decimal

Let’s try this.

Example 2-2-3 FloatingPoints.java

**01 public** **class** FloatingPoints {

02 **public** **static** **void** main(String[] args) {

03

\*04 **float** f= 1.0f/3.0f;

\*05 **double** d = 1.0/3.0;

06

\*07 System.***out***.format("%.5f \n", f);

\*08 System.***out***.format("%.3f \n", d);

\*09 System.***out***.format("there can be many values like %.3f and %.2f", f,d);

10

11 }

12

13}

explain

04 : put the result of 1/3 into float type variable f

05 : put the result of 1/3 into double type variable d

07 : Use System.out.format, print f until 5th decimal.

08 : Use System.out.format, print d until 3th decimal.

09 : Use System.out.format, print “there can be many values like “ and print f until 3th and print d until 2th decimal.

output

0.33333

0.333

there can be many values like 0.333 and 0.33

There are many different formats. %.xf is one of them. “\n” in line 7,8 means linefeed.

In line 9, there are two variables in a line. You can put multiple variables but the number of % has to be the same as the number of variables.

P8. Make a program. Pi is about 3.14159265359. when redius of a circle is 5, find the circumference of the circle and print it out to the second decimal place.

output

circumference of this circle is 31.42

[3] Boolean type

Boolean type only has two conditions. True or false. This is very useful and using less size of deta.

Let’s learn it with code.

Example 2-2-4 BooleanVariables.java

01 public class FirstJavaProgram {

02 public static void main(String[]args) {

03

\*04 boolean a = true;

\*05 System.out.println("the value of boolean variable a is : " + a);

06

\*07 boolean b = 10 > 20;

\*08 System.out.println("the value of boolean variable b is : " + b);

09

\*10 boolean c = a;

\*11 System.out.println("the value of boolean variable c is : " + c);

12 }

13

14 }

explain

04 : Put true into boolean type variable a.

05 : print a

07 : put the result of 10>20 into boolean type variable b.

08 : print b.

10 : Put the value of a into boolean type variable c.

11 : print c.

output

the value of boolean variable a is : true

the value of boolean variable b is : false

the value of boolean variable c is : true

P8. Make a program. There are three Boolean variables “juniorHighSchoolStudent”, “moreThanFiveFriends”, “ateDinner”. The value of them is going to be your answer. And print them.

output

I'm a junior high school student : true

I have more than five friends : false

I ate dinner : false

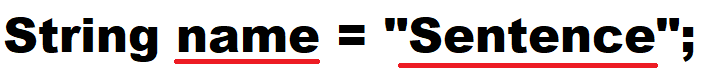
[4] String

Do you still remember your first program? The first program had only one code System.out.println(“Hello world!”);

We learned many kinds of variables. But we still didn’t have a variable that can hold “Hello world!”. Maybe char? char type only can save one letter. How can we save a sentence in a variable?

In this case, we use String type. Be careful. **S** should be written as a capital letter.

String declare and initialize



\*Underlined words are modifiable.

Meaning : put sentence into the String type variable ‘name’.

You have to write the **sentence inside double quotation marks.**

String can contain spaces and special letters too.

Example 2-2-5 StringSample.java

01 public class StringSample {

02 public static void main(String[]args) {

\*03 String a = "Hello ";

\*04 String b = "World!";

05

\*06 int i1 =1000;

\*07 int i2 =2000;

08

\*09 System.out.println(a+b);

\*10 System.out.println(a+"World!");

\*11 System.out.println(a + i1 + i2 + b);

\*12 System.out.println(i1+i2);

13 }

14 }

explain

03: put “Hello “ into a String type variable a.

04 : put “World!“ into a String type variable b.

06 : put 1000 into a int type variable i1.

07 : put 2000 into a int type variable i2.

09 : print a and b.

10 : print a and “World!”

11: print a and i1 and i2 and b

12 : print the result of i1+i2

output

Hello World!

Hello World!

Hello 10002000World!

3000

Let’s talk about line 11 and line 12.

The format of line 11 is String + int + int + String. If add something with String type, it means write these two variables in order. For example a+b = ab, 1+2 =12.

The format of line 12 is int + int. in this case, it calculate these two values.

String is a complicated type. There are lots of method in this type. We are going to learn useful part of String in this book.

P9. Make a program. There are two int variables and String variables. Using them, print “300 times 400 is 120000”

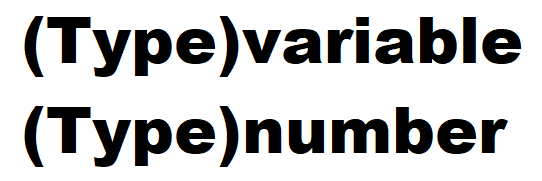
Output

300 times 400 is 120000

Section 3. Type casting.

We have learned types. Some of the types are similar. Can we change a type to a different type? The answer is yes. There is a code of type casting.

Forced cast



Meaning : cast to another type of a variable or a number(or a character).

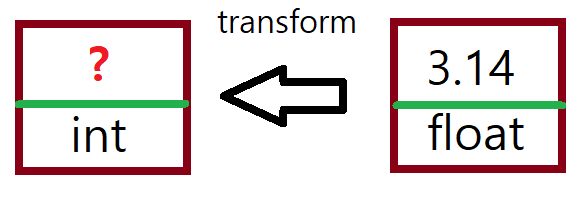
priority



**Left** type is has **lass priority** than **right type**.

If you cast to type on the right, there will be no loss of data, but if you **cast to type on the left**, you can have **loss of data**.

There is an example

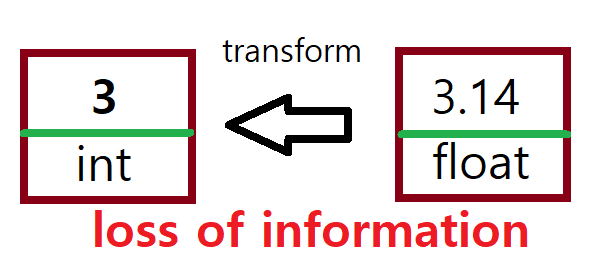


You have a float type variable. It’s value is 3.14. what if you cast it to a int type?

We know int type is an integer type. So it can’t have decimals.

So, the value in float type variable will not fit the int type. There need to be some adjustment.

Round off all the decimal



So the value will be changed to 3. There is a data loss. In mathematics, we call this situation “The range is smaller than the domain, so the value can not be stored.”

Casting left to right is called widening, and casting right to left is called narrowing.

When we are narrowing, we need forced cast, when widening, we don’t have to.

Example 2-3-1

01 **public** **class** example {

02 **public** **static** **void** main(String[]args) {

03

\*04 **byte** b = 120;

\*05 **int** i= b;

\*06 System.***out***.println("Widening : " + i);

07

\*08 **int** j = 259;

\*09 b = (**byte**)j;

\*10 System.***out***.println("Narrowing : " + b);

11

\*12 **double** d = 259.428;

\*13 i = (**int**)d;

\*14 System.***out***.println("double 259.428 to int : " + i);

15 }

16 }

04: put 120 into a byte type variable b.

05 : put b(120) into a int type variable i.

(Widening doesn’t need forced cast).

06 : print i.

08 : put 259 into a int type variable.

09 : put j(259) into byte type variable b by forced casting.

10 : print b.

12 : put 259.428 into double type variable d.

13 : put d(259.428) into int type variable i by forced casting.

14 : print i

Output

Widening : 120

Narrowing : 3

double 259.428 to int : 259

What do you think is the reason of second line of output?

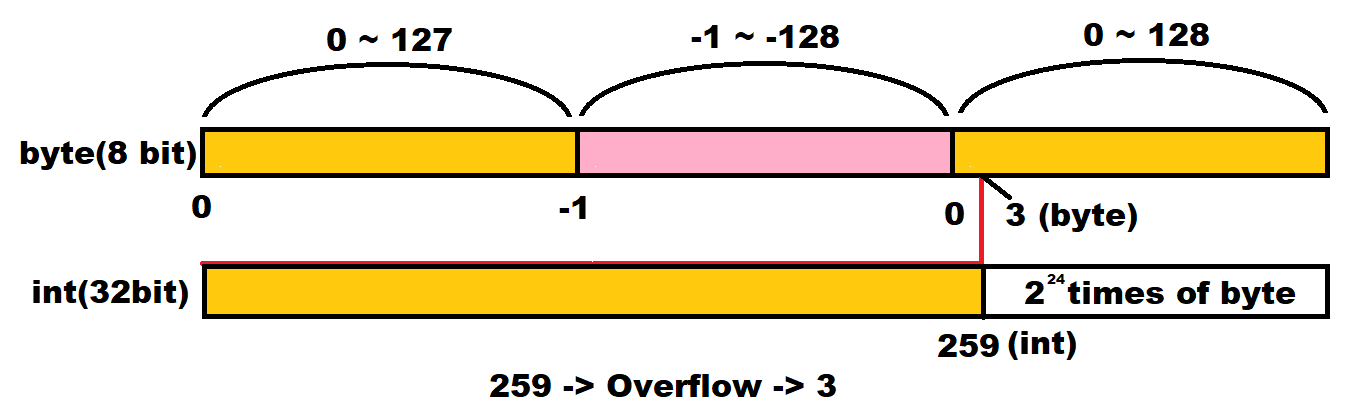
There is a clue of the reason.

Let’s go up to find chart 1. The range of byte is -128 ~ 127. But the value in j is 259. What will happen?

It is bigger than the range of the type.

So, we get wrong number. We call**“Overflow”**

This chart shows the mechanism of the overflow.



Byte can contain 256 numbers( -128 ~ 127)

259 - 256 = 3. So 259 turn one wheel and leave 3. So after overflow, we get 3.

Overflow is one of the most common errors.

P10. Make a program. Last time, we calculated circumference of a circle. This time, PI is 3.14159265359. Radius of a circle is 5. You are going to calculate the area of the circle and put it in a double type variable. You are not going to print decimals. (convert to int)

Output

78

-casting with String type

String can contain many different types. Can contain integer, character, floating point and so on.

String type can’t be changed with forced casting. There are special ways to cast.

We are going to start with casting between int and String

String to int, int to String



\*Underlined parts are variables of each type

First line casts from String to int.

Second line casts from int to String.

What if you want to add two numbers in String type? Ex) String a(“130”) and String b(“382”)

Because the type of these two numbers are String, if you do a + b, you will get “130382”

To get 130 + 382 = 512, You have to cast String to int.

example 2-3-2 StringToInt.java

01 **public** **class** StringToInt{

02 **public** **static** **void** main(String[]args) {

03

\*04 String a1 = "130";

\*05 String b1 = "382";

\*06 System.***out***.println("(String)a+b=" + (a1+b1));

07

\*08 **int** a2 = Integer.*parseInt*(a1);

\*09 **int** b2 = Integer.*parseInt*(b1);

\*10 System.***out***.println("(int)a+b=" + (a2+b2));

11 }

12 }

explain

04,05 : put “130” and “382” into String type variable a1 and b1.

06 : Print String a1 + String b1.

08,09 : Cast a1 and b1 to int type and put them into int type variable a2 and b2.

10 : print int a2 + int b2

output

(String)a+b=130382

(int)a+b=512

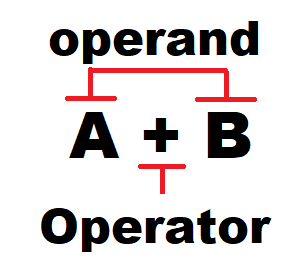
P11. Make a program. There are two int type variables. And value of them are 100 and 200. You are going to print String “100200” using these variables.

Output

100200

Section 4. Equation

How many operators do you know? + - \* / and more? There are much more operators in java programing. Generally, An equation has one operator and two operands like this picture.



There are 2 different ways to classify equations. One of them is classify by the number of operands and the other one is classify by the type of operators.

-Types of operators

[1]Arithmetic operator

Basic models of arithmetic operator are plus, minus, multiplication and division. There are more ways to express equation to make code efficiently.

The Arithmetic operators below are provided by Java.

Chart 4 Arithmetic operator

|  |  |  |  |
| --- | --- | --- | --- |
| operator | Way to use | explanation | remarks |
| + | op1 + op2 | Op1 increase op2 | Unary/Binary |
| - | op1 - op2 | Op1 subtract op2 | Unary/Binary |
| \* | op1 \* op2 | Op1 times op2 | Binary |
| / | op1 / op2 | Op1 divided by op2 | Binary |
| % | op1 % op2 | The rest of op1 divided by op2 | Binary |
| ++ | var ++  ++var | Evaluate the value before the value of var increase 1  Evaluate the value after the value of var increase 1 | Unary  Unary |
| -- | var--  --var | Evaluate the value before the value of var subtract 1  Evaluate the value after the value of var subtract 1 | Unary  Unary |

++ and -- are only for variables

This example will help you to understand how to use Arithmetic equations

Example 2-4-1 Operators.java

01public class Operators {

02 public static void main(String[]args) {

03

04

\*05 int a=6, b=4;

06

\*07 int sum = a+b;

\*08 System.out.println("a+b=" + sum);

09

\*10 int sub= a-b;

\*11 System.out.println("a-b=" + sub);

12

\*13 int mul = a\*b;

\*14 System.out.println("a\*b=" + mul);

15

\*16 int div = a/b;

\*17 System.out.println("a/b=" + div);

18

\*19 int c = ++a;

\*20 System.out.println("(pre-increase)c=" + c);

\*21 System.out.println("(pre-increase)a=" + a);

22

\*23 int d = b++;

\*24 System.out.println("(post-increase)d=" + d);

\*25 System.out.println("(post-increase)b=" + b);

26 }

27}

Explain

05 :put 6 and 4 into int type variables ‘a’ and ‘b’.

07 : put the result of a+b into int type variable ‘sum’.

10 : put the result of a-b into int type variable ‘sub’.

13 : put the result of a\*b into int type variable ‘mul’.

16 : put the result of a/b into int type variable ‘div’.

19 : after prefix increment equation on a, put it into int type variable c.

23 : after postfix increment equation on b, put it into int type variable d.

output

a+b=10

a-b=2

a\*b=24

a/b=1

(pre-increase)c=7

(pre-increase)a=7

(post-increase)d=4

(post-increase)b=5

The points here are line 19 and 23.

++ means plus one but the timing of increment is different.

a = ++b : add 1 to b. after adding, put the value of b into a

a = b++ : put the value of b into a. After that, add 1 to b.

this operator is pretty confusing but if you use this operator in right way, it will be very useful. This operator is only in programing. Which means, it is a necessary operator.

P12. Estimate the output.

**public** **class** example {

**public** **static** **void** main(String[]args) {

**int** a = 5;

**int** b = a++;

**int** c = ++a;

**int** d = b++;

**int** e = d++ + ++a;

System.***out***.println(a+", " + b + ", " + c + ", " + d + ", " + e);

}

}

[2]Relational operator

Just like the name of the operator, this operator compares operands and find out the relation of them. There are only two ways to tell the relation. True or false. If each of operand has different type, it will automatically change the type which has less priority to the higher one.

These Relational operators below are provided by Java.

Chart 5 : Relational operator

|  |  |  |
| --- | --- | --- |
| operator | Way of using | description |
| > | op1 > op2 | If op1 is bigger than op2 |
| >= | op1 >= op2 | If op1 is bigger or equal than op2 |
| < | op1 < op2 | If op1 is smaller than op2 |
| <= | op1 <= op2 | If op1 is smaller or equal than op2 |
| == | op1 == op2 | If op1 is equal to op2 |
| != | op1 != op2 | If op1 is not equal to op2 |
| instanceof | op1 instanceof op2 | If op1 is an instance of op2 |

The point there is “==”.

In programing = not means equal.

a = b : put b into a

a == b : is the value of a is equal to b? (return true or false)

This is an example of using Relational equation

example 2-4-2 relationOperator.java

01 public class relationOperator {

02 public static void main(String[]args) {

03

\*04 float aHeight = 166.5f;

\*05 float bHeight = 171.3f;

06

\*07 boolean flag;

08

\*09 flag = aHeight > bHeight;

\*10 System.out.println("Is a higher than b? : " + flag);

11

\*12 int aAge = 15;

\*13 int bAge = 15;

14

\*15 flag = aAge == bAge;

\*16 System.out.println("Are a and b same age? : " + flag);

17 }

18}

explain

03 : put 166.5 into float type variable ‘aHeight’

04 : put 171.3 into float type variable ‘bHeight’

06 : Declare Boolean type variable ‘flag’

08 :put the result of aHeight > bHeight into ‘flag’.

11,12 : put 15 into int type variables ‘aAge’ and ‘bAge’.

14 : put the result of aAge == bAge into flag.

Output

Is a higher than b? : false

Are a and b same age? : true

The point here is that the output is only “true” or “false”.

P13. Make a program. Int type variable a is 382 \* 952. Int type variable b is 584 \* 592. using Relational equation, find the higher number and print it.

Output

363664

[3] Logical operator

Logical operators evaluate the value and express it with true and false.

There are 3 operators.

and(&&) or(||) not(!).

What does ‘and’ mean? And what does ‘or’ mean?

For example) “Apple and beef are vegetables.” Does it make sense? Because of ‘beef’ does not make sense. Which means, this sentence is false. Both of elements has to be true to make the sentence true.

For example) “Apple or beef is vegetable.” Does it make sense? Or means one of them. So it make sense.

In Java, people expressed these logics in different way but has same meaning.

Chart 6 Logical operator (or, and operator)

|  |  |  |  |
| --- | --- | --- | --- |
| X | Y | X||Y | X&&Y |
| True | true | true | false |
| True | false | true | False |
| false | true | true | false |
| false | false | false | true |

Not operator

|  |  |
| --- | --- |
| X | !X |
| True | false |
| false | true |

Way to understand this chart : When X is true, Y is true, X or Y equals true, X and Y equals false.

When X is true, not X is false.

Here is a point. Logical operator **can’t calculate a value except** true or false.

There are some examples

Think ok each of them make sense or make an error.

boolean flag;

1. flag = 30 < 20;
2. flag = 30 && 20;
3. flag = 30 < 20 || 50 > 20;
4. flag = 30 < 20 && 80;
5. flag = 20 + 30 < 20 \* 2

[4] bit operator : ACS Theory Lesson 6 helps to understand this part.

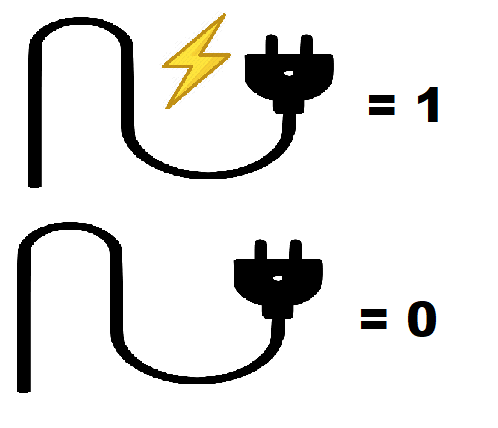
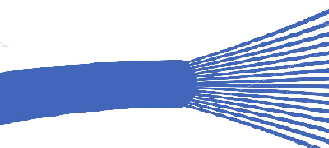
We need to learn ‘binary number’ before we start bit Operator.

People generally have 10 fingers. And we use 10 numbers from 0 to 9 (0,1,2,3,4,5,6,7,8,9) we call it decimal number. But the computer doesn’t understand decimal number because it only can read 0 and 1. Why?

This is related to the mechanism of the computer. Everybody knows the computer uses electricity and computer is connected with cables.

If there is no electricity, the computer recognize it as 0,

If there is electricity, the computer recognize it as 1.

Picture 1 Picture 2

In picture 2. It describes there are many different cables in a cable. Each of the cable delivers the information of 0 or 1. The computer combine these 0s and 1s to make other numbers.

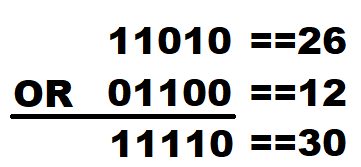
Bit operator calculates binary numbers with and(&), or(|), exclusive or(^) gates.

Chart 7 Bit Logical Operator

|  |  |  |
| --- | --- | --- |
| Operators | Way to use | example |
| & | 5&7 | 5(101)AND7(111) = 101 |
| | | 5|7 | 5(101)OR7(111) = 111 |
| ^ | 5^7 | 5(101)XOR7(111) = 010 |

The way to solve:

It is just same as using logical operator in each digit of binary number.



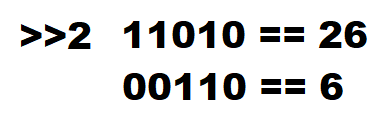
Calculation with bit logical operator (or)

[4]-2 Shift operator

Shift operator if a special operator. Move each digit of the binary number. Only integer type can be operated.

Chart 8 Shift operator

|  |  |  |
| --- | --- | --- |
| 연산자 | 사용법 | 설명 |
| << | a<<n | Convert a to binary and shift n times to the left. |
| >> | a>>n | Convert a to binary and shift n time to the right. |
| <<< | a>>>n | Convert a to binary and shift n times to the right. If a is negative, convert to positive. |



The way shift operator works

Not only binary numbers, sometimes we need to use octal and Hexadecimal numbers. Let’s learn how to describe these types.

Octal: express with 8 numbers (0 ~ 7)

Hexadecimal: express with 16 numbers (0 ~ 7 -> 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F)

|  |  |  |
| --- | --- | --- |
| types | The way to describe | Example |
| octal | Put 0 before number | 0324 |
| Hexadecimal | Put 0x before number | 0x324 |

When you print them, you will get decimal number of that value.

We can convert decimal numbers to octal, hexadecimal and binary numbers.

Convert a decimal number to a binary number.



For octal, Integer.toOctalString(decimal number), for hexadecimal, Integer.toHexString(Decimal number).

\*Underline part is modifiable.

This is called a ‘method’. A method has a function. When these skills, I will print binary, octal, hexadecimal numbers in this example

example 2-4-3 BinaryToHexadecimal.java

01 public class BinaryToHexadecimal {

02 public static void main(String[]args) {

03

\*04 int a = 100;

\*05 int b = 0100;

\*06 int c = 0x100;

07

\*08 System.out.println("Decimal number 100=" + a);

\*09 System.out.println("Octal number 100="+b);

\*10 System.out.println("Hexadecimal number 100="+c);

\*11 System.out.println("convert decimal number 100 to binary number :

12 "+Integer.toBinaryString(100));

13 }

14 }

explain

04 : put 100 into int type variable ‘a’.

05 : put 100 into int type variable ‘b’.

06 : put 100 into int type variable ‘c’.

08 : print decimal number 100.

09 : convert decimal number 100 to octal number and print.

10 : convert decimal number 100 to hexadecimal number and print.

11 : convert decimal number 100 to binary number and print.

output

Decimal number 100=100

Octal number 100=64

Hexadecimal number 100=256

convert decimal number 100 to binary number : 1100100

P13. Make a program. There is a number 492(decimal) convert it to binary, octal, hexadecimal numbers.

output

492 in binary : 111101100

492 in octal : 754

492 in binary : 1ec

Let’s focus on the way to convert to binary.

[5] Assignment operator

This is a special operator. It simplified the way of expressing equation to be more efficient.

Let’s compare the original equation and equation with assignment operators.

Chart 9 assignment operator(compare)

|  |  |
| --- | --- |
| Original Equation | Equation with assignment operator |
| j = j + 5; | j += 5; |
| j = j – 8; | j -= 8; |
| j = j \* 10; | j \*= 10; |
| j = j / 12; | j /= 12; |

Both of equations have same meaning. When we use assignment operator, we write less codes.

There are some more assignment operators

Chart 10 assignment operator

|  |  |  |
| --- | --- | --- |
| operator | Way to use | meaning |
| += | op1 += op2 | op1 = op1 + op2 |
| -= | op1 -= op2 | op1 = op1 - op2 |
| \*= | op1 \*= op2 | op1 = op1 \* op2 |
| /= | op1 /= op2 | op1 = op1 / op2 |
| %= | op1 %= op2 | op1 = op1 % op2 |
| &= | op1 &= op2 | op1 = op1 & op2 |
| |= | op1 |= op2 | op1 = op1 | op2 |
| ^= | op1 ^= op2 | op1 = op1 ^ op2 |
| <<= | op1 <<= op2 | op1 = op1 << op2 |
| >>= | op1 >>= op2 | op1 = op1 >> op2 |
| >>>= | op1 >>>= op2 | op1 = op1 >>> op2 |

I’m not telling you to memorize all these operators. You will memorize it automatically if you used them many times. You can find this chart whenever you need to use one of them.

This is an example of assignment equation

Example 2-4-4 ShortenOperator.java

01 public class ShortenOperator {

02 public static void main(String[]args) {

03

04 int a= 5;

05

\*06 a+=5;

07 System.out.println("a = " + a);

08

\*09 a%=4;

10 System.out.println("a = " + a);

11

\*12 a <<=4;

13 System.out.println("a = " + a);

14

15 boolean b = false;

16

\*17 b &= a>2;

18 System.out.println("b = " + b);

19

\*20 b|= a>2;

21 System.out.println("b = " + b);

22 }

23}

Explain

06 : do a= a+5

09 : do a = a%4

12 : do a= a<<4

17 : do b = b & a>2

20 : do b = b | a>2

output

a = 10

a = 2

a = 32

b = false

b = true

P14. Make a program. There is an int type variable containing 200. This number will be increased by 482\*92-321.

Print the variable after this calculation using shortened operator.

Output

44223

5) Ternary operator

Java has an operator that has 3 operands.



\*A and B and C are equations

Meaning: if A is true, return B, if A is false, return C

Just see this code real quick and understand it

Example 2-4-5 TernaryOperator.java

01 public class result {

02 public static void main(String[]args) {

03

\*04 int a = 10>5?3:5;

\*05 Boolean b = 100==101?true:false;

06

07 System.out.println(a + " , " + b);

08 }

09}

Explain

04 : put ‘3’ into a if 10>5 is true, else put ‘5’ into a

05: put ‘true’ into b if 100==101 is true, else put ‘false’ into b

output

3 , false

P15. Make a program. This program has two variables. The values of each of them are 30, 29. Using ternary operator to find which variable has bigger value.

Output

30

We learned every kind of operators. There is something we have to know at the end. What if we use many different kinds of operators in once? What would be the order of calculation?

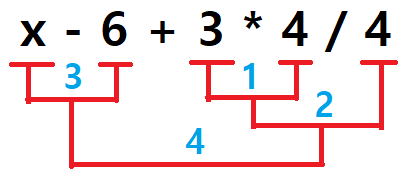
In normal math calculation 2+3\*4, we multiply first and add 2. The reason is that multiplication has higher priority than increment. There are priority for each of operators.

Chart 11 Priority of Operators

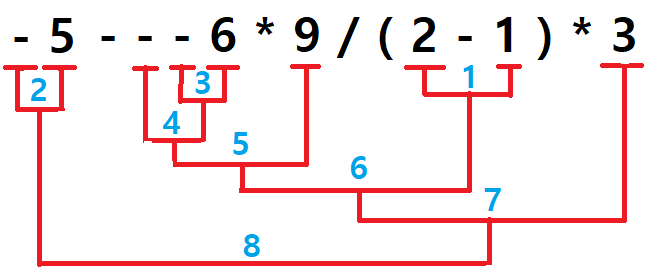
|  |  |  |  |
| --- | --- | --- | --- |
| Priority(1 <- highest) | operator | example | name |
| 1 | [] | a[b] | Index Operator |
| () | (a+b)\*c | Bracket |
| . | a.b | Member access operator |
| 2 | ++ | ++a, a++ | Increment operator |
| -- | --a, a-- | Decrement operator |
| + | +a | Unary + operator |
| - | -a | Unary – operator |
| ! | !a | NOT operator |
| ~ | ~a | Bit complement operator |
| 3 | New | new class\_name | New operator |
| () | (short) | Cast operator |
| 4 | \* | a \* b | Multiply operator |
| / | a / b | Divide operator |
| % | a % b | Rest operator |
| 5 | + | a + b | + operator |
| - | a - b | - operator |
| 6 | << | a << b | Shift operator |
| >> | a >> b |
| >>> | a >>> b |
| 7 | < | a < b | Relational operator |
| > | a > b |
| <= | a <= b |
| >= | a >= b |
| instanceof | a instanceof b |
| 8 | == | a == b | Equivalent operator |
| != | a != b |
| 9 | & | a & b | Bit AND operator |
| 10 | ^ | a ^ b | Bit XOR operator |
| 11 | | | a | b | Bit OR operator |
| 12 | && | a && b | AND operator |
| 13 | || | a || b | OR operator |
| 14 | ?: | a ? b : c | Ternary operator |
| 15 | = | a = b | Assignment operator |
| \*=,/= … | a \*= b, a/= b, … | Shortened operator |

If they have same priority, calculate from left.

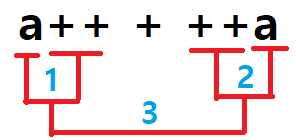
Let’s practice with some equations.



Multiplication and division has higher priority then others.



Bracket go first and calculate unary operators



Let’s say a = 10. What would be the answer?

Bonus Section(Things to know)

Identifier: the names we create. Like the name of a variable. We can name whatever we want but actually there are few rules.

1. You can use letters, numbers, and special characters (\_, $)
2. You can start the identifier with letters and special characters but not numbers.
3. Reserved words are not available for an identifier.
4. True, false, null can’t be an identifier.
5. Length of an identifier does not matter.
6. Upper and lower characters are different.

Number 3 has a word “Reserved words”. Reserved words mean words that are already in use in JAVA. For example, int is a reserved word meaning an integer type. So ‘int’ can be an identifier.

There are several reserved words in Java

Chart 12. reserved words

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Abstract | Const | Finally | Interface | Short | Transient | Assert | Continue | Float | Long |
| Static | Try | Boolean | Default | For | Native | Strictfp\* | void | Break | Do |
| Goto | New | Super | Volatile | Byte | Double | If | Package | Switch | While |
| Case | Else | Implements | Private | Synchronized | Catch | Enum | Import | Protected | This |
| Char | Extend | Instanceof | Public | Throw | Class | Final | Int | Return | throws |

And there are some more extra rules from programmers. The first letter of the name of a variable need to be lower case, the first letter of a class need to be upper case, the name of a method has to be lower case. Even if you don’t do this, you will not get any errors. For understand other people’ code, programmers made their own rules.

Section 5. Input

All of our example codes were only for printing something. What if we give some input to computer, will the computer understand the input? Yes it will. To give some input, We need to learn two more things that we never talked about. They are “header” and “generate an object”. These parts are very tough to understand now so we are not going to talk about these two here. We are going to say these two parts of code are necessary to write to make input. So just memorize these parts of codes.

**Example 2-5-1 input1.java**

**\*01 import** java.util.Scanner;

**02 public** **class** input1 {

03 **public** **static** **void** main(String[]args) {

04

05 **int** a;

06 System.***out***.print("write a number : ");

07

\*08 Scanner sc= **new** Scanner(System.***in***);

\*09 a=sc.nextInt();

10

11 System.***out***.println(" your number is " + a);

12

13 }

14 }

explain

01 : declare that we are going to use the library called java.util.Scanner.

08 : Declare an object of Scanner is sc.

09 : get an int type input and save it in a variable called ‘a’.

Input output

write a number : 5 your number is 5

P16. Make a program. The price of a ticket is 5 dollars. This program will ask a user “how many tickets you need?” and get the input. The program will tell the user total price.

Output

how many tickets do you need? : 5

the total price is 25dollars.

To make an input, 01, 08, 09 lines of code are essential. Just write 01 and 08 line if you want to make input(we are not going to talk about these lines). But you have to understand line 09 now. Input can be anything that you can type. But computer needs to save your input to a specific type. Ex) int, char, String, and so on. You can tell computer what kind of input the user will give. There are methods to get each type of input below.

nextBoolean() : the type of the next input is Boolean.

nextDouble() : the type of the next input is double.

nextFloat() : the type of the next input is float.

nextInt() : the type of the next input is int.

nextLine() : the type of the next line of input is String.

nextLong() : the type of the next input is long.

nextShort() : the type of the next input is short.

next().charAt(0) : the type of the next input is char.

This example shows how to use these examples

example 2-5-2 input2.java

**\*01 import** java.util.Scanner;

**02 public** **class** input2 {

03 **public** **static** **void** main(String[]args) {

04

05 String s;

06 **int** a;

07 **char** c;

08 **double** d;

09

\*10 Scanner sc= **new** Scanner(System.***in***);

11

12 System.***out***.print("write your name : ");

\*13 s=sc.nextLine();

\*14 System.***out***.println("your name is " + s);

15

16 System.***out***.print("write your ager : ");

\*17 a=sc.nextInt();

\*18 System.***out***.println(" you are " + a + "years old");

19

20 System.***out***.print("write a letter : ");

\*21 c=sc.next().charAt(0);

\*22 System.***out***.println("your letter is " + c);

23

24 System.***out***.print("write the number of pi as much as you know :

25 ");

\*26 d=sc.nextDouble();

\*27 System.***out***.println("pi is " + d);

28 }

29}

Explain

01,10 : Essential codes for input.

13 : Get a line of input and save it in ‘s’.

14 : Print s.

17 : Get an int type of input and save it in ‘a’.

18 : Print a.

21 : Get an char type of input and save it in ‘c’.

22 : Print c.

26 : Get an double type of input and save it in ‘d’

27 : Print d.

input

write your name : Aaron kim

write your ager : 15

write a letter : A

write the number of pi as much as you know : 3.14

output

your name is Aaron Kim

you are 15years old

your letter is A

pi is 3.14

Except nextLine(), space(“ ”) can separate the input.

For example)

int a = sc.nextInt();

int b = sc.nextInt();

If your input is 10 11 as below

Input

10 11

The value of ‘a’ will be 10 and the value of ‘b’ will be 11.

Tips for being a better programmer.

We make program with other people. Which means your code needs to be good for reading. There are some tips for it.

1. Way of naming an identifier.

There are extra rules from programmer. The first letter of variables and methods has to be lower case and the first letter of a class has to be upper case. If an identifier consists of more than 2 words, the first letter of words has to be upper case. A constant has to be upper case.

Ex1) class Overwatch : class

Ex2) int age : variable

Ex3) int ageOfSister : variable with 3words

Ex4) define PI 3.14 : constant

1. Spacing and changing lines

Spacing and changing line never matters to a program. It would be a great way to make the code better to read.

Good way of Spacing : equation

Ex) int i=1+1;

* int i = 1 + 1; (space between each operators and operands)

boolean b=(3.04\*49+62)>=(23-4)/92%3\*92;

* boolean b = (3.04 \* 49 + 62) >= (23 - 4) / 92 % 3 \* 92;

Good way of changing line : separate to parts

In a program, there are many different type of codes like declare, calculation, input and output and so on. We separate each part by changing line.

Ex)

int a = 10;

float f = 3.22f;

(change line)

f+=a;

a++

(change line)

System.out.println(“numbers : “ + a + f);

1. annotation

I want to take a not about the code, but if I write it on the code, there will be errors. If you use annotation codes, your compiler will not recognize them as a part of code. It’s only going to think they are memos.

There are 3 ways of making annotation.

1. //…….. : annotate a line.
2. /\*……..\*/ annotate from the start to an end.
3. /\*\*……..\*/ : annotate several lines and make description following Javadoc.

Example of using annotation.

Example Tip-1 java

\*01 //booking program

02 import java.util.Scanner;

03 public class example {

04 public static void main(String[]args) {

05

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

07

08 System.out.println("If you want to book, tell me your name : ");

09

10 String name = sc.nextLine(); //get input in String type

11

\*12 System.out.println(name + ", you made a reservation.");

13

\*14 /\*

\*15 System.out.println("have a nice day!");

\*16 System.out.println("see you next time!");

\*17 \*/

18

19 }

20 }

Explain

01 : line 01 is annotated

12 : in line 12, after // is annotated.

14~17 : line 14 ~17 are annotated. “Have a nice day!” And “see you next time” won’t be printed.

Way of using annotation 1: if I want to save code

You made two algorithm to solve one problem. If you want to try one of your algorithm, you can make annotation to the other algorithm. So, you don’t need to erase your code.

Way of using annotation 2: make explanation

Sometimes, it’s hard to understand my own code. If you write explanation on that part, it will be easier to organize and understand the code.

Way of using annotation 3: if there is errors/ find errors

If there is an error in your code, you can use annotation the suspicious part of your code and try other ways

Practice

Common sense

What is the default type of an integer and floating point?

Figure out the identifiers are acceptable or not

String first name = “Aaron”; ( )

int child-age = 10; ( )

double \_distance = 35.92; ( )

1. Predict the output

A-2-1

**public** **class** example {

**public** **static** **void** main(String[]args) {

**float** f=100.32f;

**int** a;

**char** c;

a=(**int**)f;

c=(**char**)a;

f=a;

System.***out***.println(a);

System.***out***.println(c);

System.***out***.println(f);

}

}

A-2-2

**public** **class** example {

**public** **static** **void** main(String[]args) {

**byte** b = 100;

**int** i;

i= b;

System.***out***.println(i);

i= 300;

b=(**byte**)i;

System.***out***.println(b);

}

}

A-2-3

**public** **class** example {

**public** **static** **void** main(String[]args) {

**int** a=10;

**int** b=5;

++a;

System.***out***.println(a);

b=a++;

System.***out***.println(b);

System.***out***.println(a);

a=++b;

System.***out***.println(a);

++b;

a++;

System.***out***.println(a+b);

}

}

A-2-4

**public** **class** example {

**public** **static** **void** main(String[]args) {

**int** days = 30;

String subject = "math";

**float** score = 89.5f,scoreForA = 90.0f;

**char** grade = 'A';

System.***out***.println("I studied "+subject+" for "+days+"days, and I got "+score+" on this test");

System.***out***.println("it was so close. I needed "+(scoreForA-score)+"more points to get " + grade);

}

}

A-2-4

public class result {

public static void main(String[]args) {

int x = 1;

System.out.println((true) && (3>4));

System.out.println(!(x>0));

System.out.println((x!=1)||(x<1));

System.out.println((x>=0)||!(x==1));

}

}

A-2-5

public class result {

public static void main(String[]args) {

System.out.println(4<<2);

System.out.println(-4<<2);

System.out.println(3&2);

System.out.println((3^2));

System.out.println((3|2));

}

}

A-2-6

public class result {

public static void main(String[]args) {

int a=3,b=2,c=1,d=4,e=5;

int sum1= a\*b+d/c%e;

int sum2= -a \* (b+b) - c++ \* d / (4 - 2) \* e;

int sum3= (a - b) \* 10 % 3 + --c \* d - (4 - 2);

System.out.println("sum1 = " + sum1);

System.out.println("sum2 = " + sum2);

System.out.println("sum3 = " + sum3);

}

}

1. Find error if there are

B-2-1

public class result {

public static void main(String[]args) {

byte b;

int i = 300;

long j = 9876543210;

b=i;

i=j;

System.out.println(b);

Sytem.out.println(i);

Sytem.out.println(j);

}

}

B-2-2

public class result {

public static void main(String[]args) {

int i = 30>25;

boolean b1 = 42\*36||24\*19;

boolean b2 = 24>2&&39%3<3;

float f = (byte)39+20;

String s = "1231412 + 3291321 /482332";

}

}

1. Build a Program

C-2-1

I’m trying to make a registration program in a game company. Get a user’s profile and print their information.

What’s on their profile : name, age, e-mail address, phone number.

Result

Name : Aaron Kim

Age : 15

E-mail : [Aaron@zmail.com](mailto:Aaron@zmail.com)

Phone : 4082839123

Aaron kim, 15 years old, [Aaron@zamil.com](mailto:Aaron@zamil.com), 4082839123

C-2-2

Before building a structure, a building company needs to find out the volume of that place. The building designer Sam who built thousands of buildings doesn’t want to calculate volume of the place anymore. So, he planned to make a program that calculates the volume if a user writes width, length, height. Make this program

Result

Width : 200

Length : 200

Height : 400

Volume : 16000000

c-2-3

ACE Pizza restaurant wants to provide a differentiated service to customers. So They decided to make pizza in sizes that the customers want. Not like small, regular, large size. Get the radius of the pizza from customer and make that size of pizza. Make a program that get radius from customer and tell them the area of the pizza and the price of the pizza. (1cm per 1 dollar) Pi is 3.14

Result

Welcome to ACE Pizza!

Tell me the radius of your pizza: 10

The area of your pizza is 314cm2

It is 10 dollars.

c-2-4

We are trying to make an ATM machine. ATM give the money in the most efficient way. Make a program that get the amount of the money and calculates the best way to give. \*there are only 100,50,20,5,1 dollars and there is cent doesn’t count.

result

how much? : 329

100 dollars : 3

50 dollars : 0

20 dollars : 1

5 dollars : 1

1 dollars : 4