HOLIDAY HOMEWORK 假期作业

AP Computer Science A

Exercise sheets, topic by topic.

每个单元：练习卷。

For class or self-study • no answer keys included.

Contents 目录

1 Variables and data types 3

2 Making decisions - if and boolean 9

3 Writing a class 15

4 Arrays 21

1 Variables and data types – Part 1

Name: _____ Class: _____ Date: _____

Vocabulary 词汇

Write each word three times. 把每个单词抄写三遍。

English	中文	Write it 3 times (English)
variable	变量	
data type	数据类型	
expression	表达式	

Recall

You may have written a little code before, or used a calculator app:

- A program stores values and does calculations with them.
- Numbers can be whole (like 5) or decimal (like 2.5).
- Text and numbers are different kinds of data.

This sheet lines up variables and data types in Java. Each idea has a worked example.

New ideas

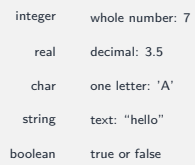
1 Variables

A **variable** 变量 is a named box that stores a value. You give it a type and a name, then a value:

```
int age = 16;
double price = 2.5;
```

2 Data types

A **data type** 数据类型 says what kind of value a variable holds:



- `int` –a whole number,
- `double` –a decimal number,
- `boolean` –`true` or `false`,
- `String` –text.

3 Expressions

An **expression** 表达式 combines values with operators (`+ - * / %`) to compute a result.

Worked example. `3 + 4 * 2` is `11` (multiply before add), and `10 % 3` is `1` (the remainder).

4 Integer division

When you divide two whole numbers, Java throws away the fractional part:

Worked example. `7 / 2` is `3` (not `3.5`), but `7.0 / 2` is `3.5` (a decimal forces real division).

Watch out: dividing two `int` values **truncates** –`5 / 2` gives `2`, not `2.5`. To get a decimal, make one value a `double` first.

Practice

Try these in order. The methods are all above.

2.1 State what data type you would use for (a) a person's age, (b) their height in metres, (c) whether they passed. [3]

2.2 Evaluate `2 + 3 * 4`. [2]

2.3 Evaluate `9 / 2` and `9 % 2`. [2]

2.4 Evaluate `9.0 / 2`. [1]

2.5 Explain why `1 / 2` gives `0` in Java. [2]

1 Objects, Strings, and methods – Part 2

Name: _____ Class: _____ Date: _____

Vocabulary 词汇

Write each word three times. 把每个单词抄写三遍。

English	中文	Write it 3 times (English)
object	对象	
method	方法	

Recall

In Part A you met variables and types. Now meet **objects** and how to use them:

- An object is a ”thing” in a program, like a piece of text (a `String`).
- You do things to an object by calling its methods.

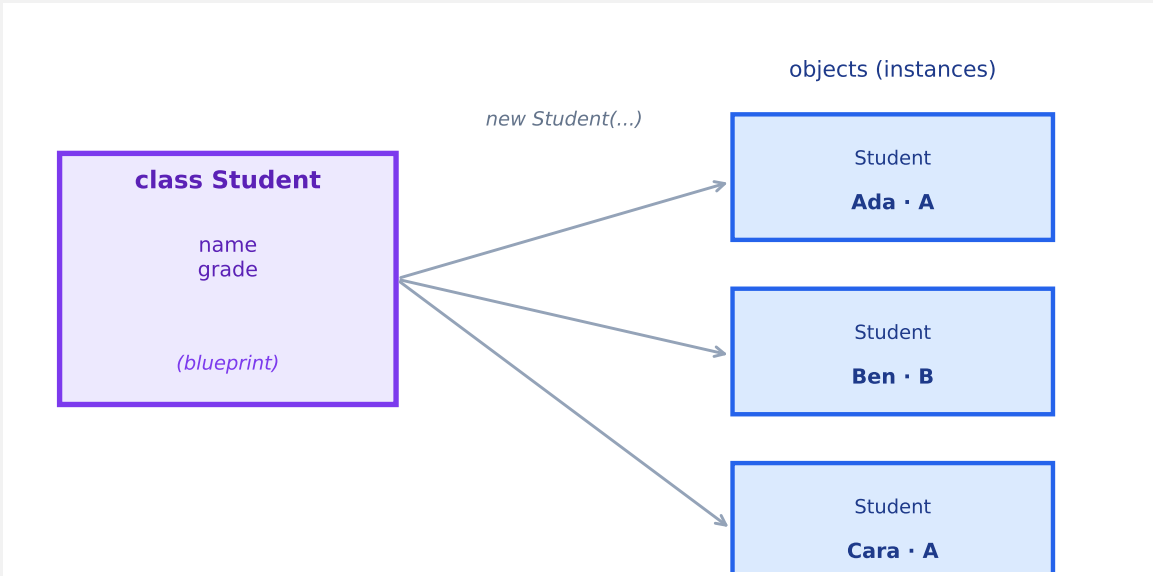
Each idea has a worked example before the Practice.

New ideas

1 Objects

An **object** 对象 bundles data with actions (methods). You create one with `new`:

```
Scanner in = new Scanner(System.in);
String s = "COMPUTER";
```



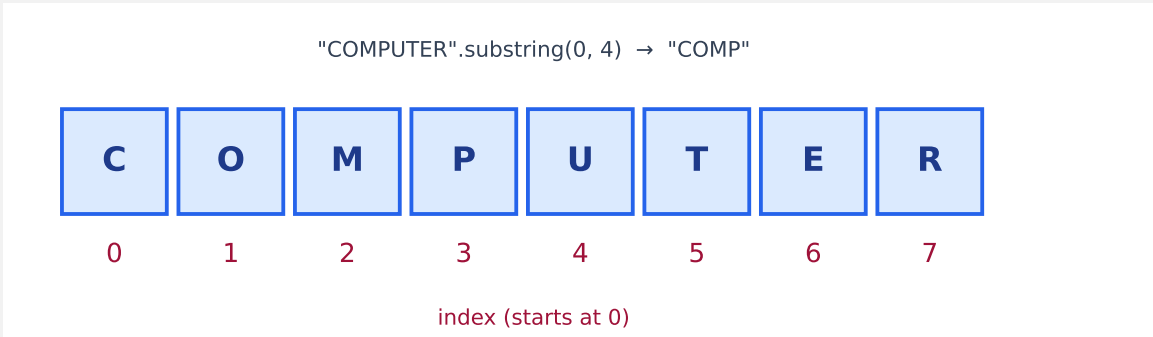
2 Calling a method

A **method** 方法 is an action you run on an object, written `object.method(...)`:

```
s.length();       // how many characters
s.substring(0,4); // part of the text
```

3 Strings

A `String` is text, and its characters are numbered from `0`.



Worked example. For `String s = "COMPUTER"`: `s.length()` is 8; `s.substring(0,4)` is `"COMP"` (characters `0,1,2,3`, index `4` excluded).

4 Comparing text

Use `.equals(...)` to compare Strings, not `==`.

Watch out: `s.substring(a, b)` includes index `a` but **excludes** `b`. And String characters count from `0`, so the first character is index `0`, not `1`.

Practice

Try these in order. The methods are all above. Use `String s = "PROGRAM";` (indices `0`–`6`).

2.1 State what an object is, in your own words. [2]

2.2 Find `s.length()`. [1]

2.3 Find `s.substring(0,4)`. [2]

2.4 Find `s.substring(3)` (from index 3 to the end). [2]

2.5 State which character is at index `0` of `s`. [1]

2 Making decisions - if and boolean – Part 1

Name: _____ Class: _____ Date: _____

Vocabulary 词汇

Write each word three times. 把每个单词抄写三遍。

English	中文	Write it 3 times (English)
boolean	布尔	

Recall

Programs have to make decisions:

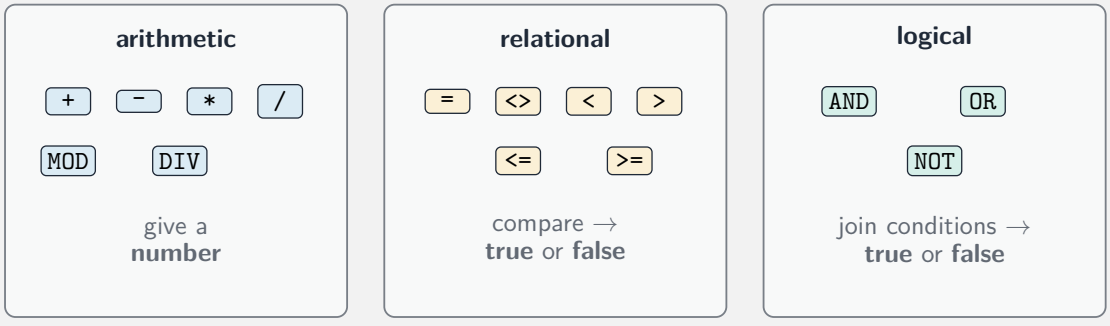
- ”If it is raining, take an umbrella.”
- A game checks if your score beats the high score.
- A test is either passed or failed –true or false.

This sheet lines up how programs decide. Each idea has a worked example.

New ideas

1 Boolean values

A **boolean** 布尔 value is either `true` or `false`. Comparisons produce booleans:



`==` (equal), `!=` (not equal), `<`, `>`, `<=`, `>=`.

Worked example. `5 > 3` is `true`; `4 == 5` is `false`.

2 The if statement

An `if` statement runs code **only when** a condition is true:

```
if (score >= 50) {
    System.out.println("Pass");
}
```

3 if / else

Add `else` to do something when the condition is false:

```
if (score >= 50) System.out.println("Pass");
else             System.out.println("Fail");
```

4 Combining conditions

- `&&` means **and** (both must be true),
- `||` means **or** (at least one is true),
- `!` means **not** (flips true and false).

Watch out: `==` compares two values, but a single `=` **assigns** a value. Writing `if (x = 5)` is a bug –use `if (x == 5)`.

Practice

Try these in order. The methods are all above.

2.1 State the two possible values of a boolean. [1]

2.2 Evaluate `7 > 4` and `3 == 4`. [2]

2.3 Evaluate `(5 > 3) && (2 > 4)`. [2]

2.4 A student has `score = 80`. State what `if (score >= 50)` prints given the if/else above. [1]

2.5 State the difference between `==` and `=`. [2]

2 Loops – Part 2

Name: _____ Class: _____ Date: _____

Vocabulary 词汇

Write each word three times. 把每个单词抄写三遍。

English	**中文**	**Write it 3 times** (English)
while loop	循环	

Recall

In Part A you met decisions. Now make the computer **repeat** work:

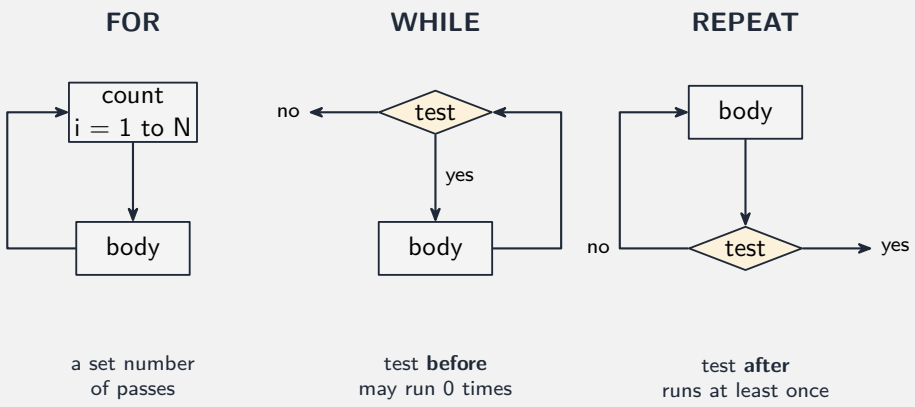
- Printing the numbers 1 to 100 by hand would be slow.
- A loop repeats the same steps many times, quickly.

Each idea has a worked example before the Practice.

New ideas

■ 1 The while loop

A **while loop** 循环 repeats **while** its condition is true:




```
int i = 0;
while (i < 3) {
    System.out.println(i);
    i++;
}
```

This prints 0, 1, 2.

■ 2 The for loop

A `for` loop packs the setup, condition, and update into one line –best when you know the count:

```
for (int i = 0; i < 5; i++) {
    // runs 5 times: i = 0,1,2,3,4
}
```

■ 3 Counting the repeats

A loop `for (int i = 0; i < n; i++)` runs exactly `n` times.

Worked example. `for (int i = 1; i <= 4; i++)` runs 4 times (for `i = 1, 2, 3, 4`).

■ 4 Nested loops

A loop **inside** another loop runs the inner one fully for each pass of the outer. If the outer runs 3 times and the inner 4, the inside runs $3 \times 4 = 12$ times.

Watch out: a loop must **change** something each time (like `i++`) so the condition eventually becomes false. If it never does, the loop runs forever –an infinite loop.

Practice

Try these in order. The methods are all above.

2.1 State how many times `for (int i = 0; i < 6; i++)` runs. [1]

2.2 State the values printed by the while loop above (`i` from `0` while `i < 3`). [2]

2.3 State how many times `for (int i = 1; i <= 10; i++)` runs. [2]

2.4 A nested loop has an outer running 5 times and an inner running 3 times. State how many times the inner body runs in total. [2]

2.5 Explain what makes a loop ”infinite”. [2]

3 Writing a class – Part 1

Name: _____ Class: _____ Date: _____

Vocabulary 词汇

Write each word three times. 把每个单词抄写三遍。

English	中文	Write it 3 times (English)
class	类	
constructor	构造函数	

Recall

In earlier sheets you **used** objects like `String`. Now you will **build your own**:

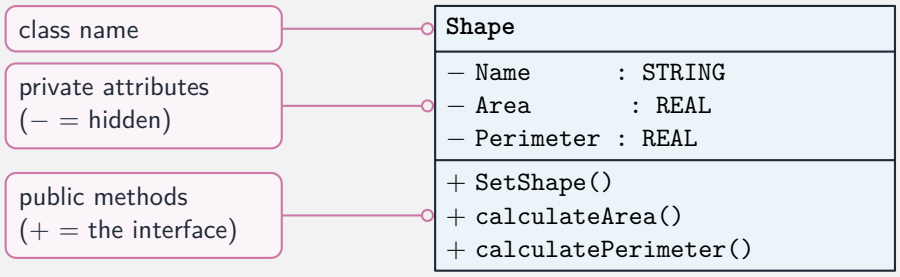
- A `String` is one type of object; you can design your own types too.
- A class is like a blueprint for making objects.

This sheet lines up how to write a class. Each idea has a worked example.

New ideas

■ 1 A class is a blueprint

A **class** 类 is a blueprint that describes what its objects hold and do. From one class you make many objects.



2 Fields

A class's **fields** are its data –the values each object stores:

```
public class Student {
    private String name;
    private int score;
}
```

3 The constructor

A **constructor** 构造函数 sets up a new object when you use `new`:

```
public Student(String n, int s) {
    name = n;
    score = s;
}
```

4 Making objects

```
Student a = new Student("Amy", 80);
```

Each `new` builds a separate object with its own `name` and `score`.

Watch out: a class is only a **blueprint** –it does nothing by itself. You must create an object with `new` before you can use it, just as a house plan is not a house.

Practice

Try these in order. The methods are all above.

2.1 State the difference between a class and an object. [2]

2.2 State what the "fields" of a class are. [1]

2.3 State what a constructor does. [2]

2.4 Write a line that creates a `Student` object called `b` with name `"Ben"` and score `70`. [2]

2.5 Explain why you can make many different objects from one class. [2]

3 Scope, methods, and references – Part 2

Name: _____ Class: _____ Date: _____

Vocabulary 词汇

Write each word three times. 把每个单词抄写三遍。

English	**中文**	**Write it 3 times** (English)
Scope	作用域	

Recall

In Part A you built a class. Now two ideas about how variables and objects behave:

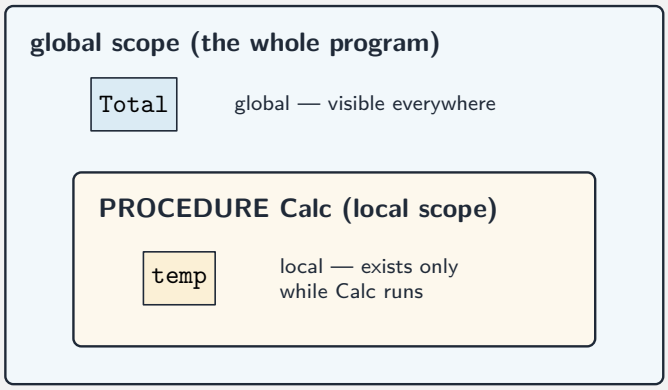
- A variable made inside a method only exists there.
- Passing an object into a method can change the original.

Each idea has a worked example before the Practice.

New ideas

1 Scope

Scope 作用域 is where a name can be used. A variable declared inside a method (a **local** variable) exists **only** inside that method.



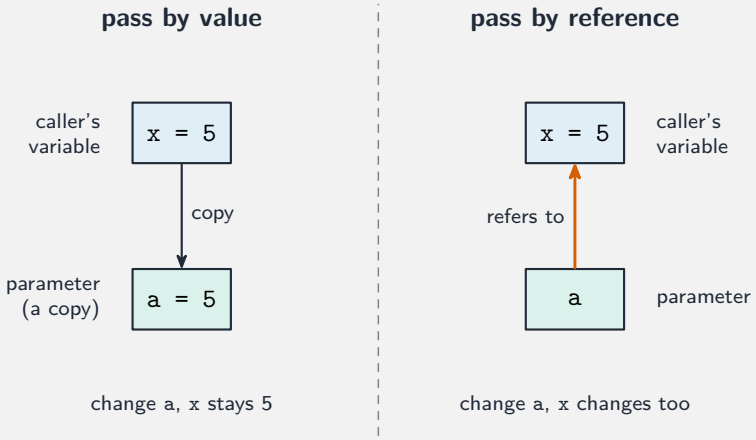
2 Methods return values

A method can compute a value and **return** it:

```
public int doubleIt(int x) {
    return x * 2;
}
```

3 Passing objects

When you pass an **object** to a method, Java passes a **reference** to it, so changes to its data are seen by the caller.



4 Passing primitives

When you pass a **primitive** (like an `int`), Java passes a **copy**, so changing it inside the method does **not** affect the caller.

Watch out: changing an object's fields inside a method **does** affect the caller (it shares the object), but changing a copied `int` does **not**. Objects are shared; primitives are copied.

Practice

Try these in order. The methods are all above.

2.1 State what ”scope” means. [1]

2.2 State where a local variable can be used. [1]

2.3 For `doubleIt` above, state what `doubleIt(6)` returns. [1]

2.4 State whether changing an object's field inside a method affects the caller, and why. [2]

2.5 State whether changing a passed `int` inside a method affects the caller, and why. [2]

4 Arrays – Part 1

Name: _____ Class: _____ Date: _____

Vocabulary 词汇

Write each word three times. 把每个单词抄写三遍。

English	中文	Write it 3 times (English)
array	数组	

Recall

A single variable holds one value, but real problems have many:

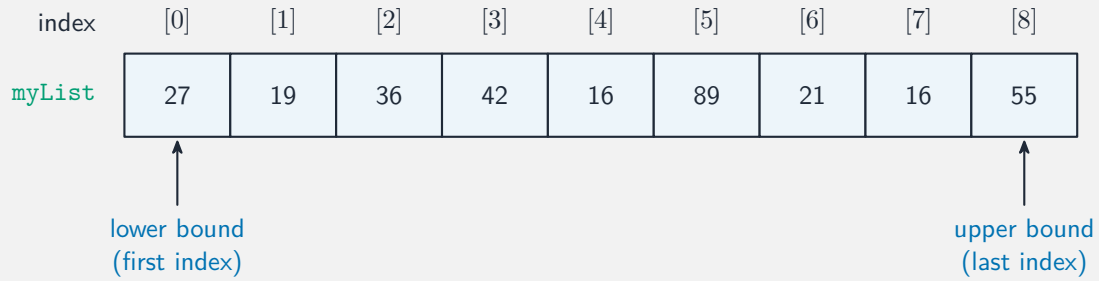
- A class has many students' scores, not just one.
- You cannot make a separate variable for every value.

This sheet lines up how to store many values in an array. Each idea has a worked example.

New ideas

1 What an array is

An **array** 数组 is a fixed-size, ordered list of values of the same type. Its positions are numbered from `0` to `length - 1`:




```
int[] vals = {3, 1, 4, 1, 5};
```

2 Reading and writing

Use the index in square brackets:

```
int first = vals[0];   // 3
vals[1] = 9;           // change index 1 to 9
int n = vals.length;   // 5
```

3 Looping through

Visit every element with a `for` loop:

```
for (int i = 0; i < vals.length; i++) {
    System.out.println(vals[i]);
}
```

4 Common tasks

Summing or finding the largest is a loop with a running result.

Worked example. To sum `{3, 1, 4}`: start at `0`, add each element → `0+3+1+4 = 8`.

Watch out: the valid indices run from `0` to `length - 1`. For an array of length `5`, `vals[5]` is **out of bounds** –the last valid index is `4`.

Practice

Try these in order. The methods are all above. Use `int[] a = {7, 2, 9, 4};`.

2.1 State the value of `a[0]` and `a[2]`. [2]

2.2 State the value of `a.length`. [1]

2.3 State the largest valid index of `a`. [1]

2.4 Find the sum of all the elements of `a`. [2]

2.5 State what is wrong with writing `a[4]`. [2]

4 Searching and sorting – Part 2

Name: _____ Class: _____ Date: _____

Vocabulary 词汇

Write each word three times. 把每个单词抄写三遍。

English	中文	Write it 3 times (English)
Linear search	线性搜索	
Binary search	二分搜索	

Recall

In Part A you stored many values in an array. Now two everyday tasks:

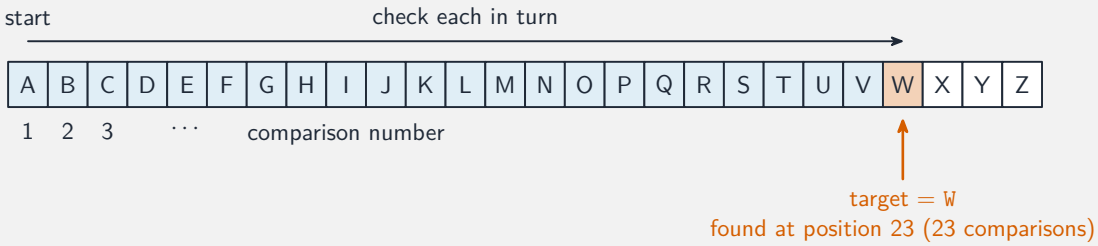
- **Finding** a value in a list.
- **Sorting** a list into order.

Each idea has a worked example before the Practice.

New ideas

1 Linear search

Linear search 线性搜索 checks each element in turn until it finds the target. It works on **any** list and takes up to n steps.



2 Binary search

Binary search 二分搜索 works only on a **sorted** list: check the middle, then throw away the half that cannot contain the target, and repeat. Each step halves the list,

so it takes about $\log_2 n$ steps.

Worked example. Search a sorted list of 8 items: $8 \to 4 \to 2 \to 1$, at most 3 comparisons –far fewer than checking all 8.

3 Sorting

Sorting puts a list in order. Simple methods like **selection sort** repeatedly find the smallest remaining value and place it next.

4 Why sorting helps

Once a list is sorted, binary search can find items much faster –which is why sorting is so useful.

Watch out: binary search only works on a **sorted** list. Running it on an unsorted list gives wrong answers –you must sort first.

Practice

Try these in order. The methods are all above.

2.1 State how linear search looks for a value. [1]

2.2 State the one thing binary search requires of the list. [1]

2.3 A sorted list has 16 items. State roughly the most comparisons binary search needs ($\log_2 16$). [2]

2.4 Explain why binary search is much faster than linear search on a large sorted list. [2]

2.5 State what would go wrong if you ran binary search on an unsorted list. [2]