

16 个很简单、可能考、看文字会学起来更快的知识点

1/16 固定开头并不是那么“固定”

本来，固定开头是：

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

其实，上面红色部分是可以改的：

- ① **Test** 可以随便写，比如写成：jinqianbao ；
- ② **args** 可以随便写，比如写成：weimengbadao ；
- ③ **[]** 可以写在 String 后，也可以移动到 String 后面的单词(args 或其他)后。

考试的时候，如果碰见哪一题固定开头和你记住的不同，不用奇怪，固定开头是可以有自己的个性的。

2/16 int 型变量必须砍掉小数部分

我们只要定义了变量是 int 型，那么，这个 int 型变量永远都不会有小数部分，它的小数部分都会被直接粗暴的砍掉。

例如，我们定义了 a 为 int 型变量，让它的值等于 5，再让它的值减半：

```
int a;  
a=5;  
a=a/2;
```

本来，a=5，再除以 2 后，取值应该是 2.5，但是，因为 a 为 int 型变量，int 型变量的小数部分会被直接粗暴的砍掉。所以，a 不能等于 2.5，后面的.5 会被粗暴的砍掉，也就是说，a=2。（注意，是粗暴的砍掉，java 不讲究四舍五入）

考试的时候，有些小题，愿意给你一个程序，让你输出运行后的结果，比较喜欢在 int 型变量这里设置小陷阱。

3/16 可以有小数的类型还有 float

Java 中带小数的数默认是 double 型的。除了 double，float 型也可以有小数。但

是要注意，让一个 `float` 型的变量等于一个带小数的数的时候，数字后面需要加一个字母 `f`，例如：

```
float a = 3.14f;
```

```
float b;
```

```
b = 2.72f;
```

考试的时候，注意小题里的小数带不带这个 `f`，带 `f` 的小数就是 `float` 型，不带的就是 `double` 型。

4/16 用 `boolean` 定义对错变量

请理解下列语句的意思。

```
boolean a;
```

【定义 `a` 为一个对错变量，也就是说 `a` 的结果要么是 `true`，要么是 `false`（`true` 和 `false` 注意都要小写）】

```
boolean a = (3 > 5);
```

【定义 `a` 为一个对错变量，也就是说 `a` 的结果要么是 `true`，要么是 `false`，它的结果到底是 `true` 还是 `false` 取决于 `(3 > 5)` 是 对 还是 错，因为 `3 > 5` 是 错 的，错 的英文单词是 `false`，所以这条语句呜呜渣渣一大堆，想表达的就是 `a` 的结果是 `false`】

考试时，看到 `boolean` 语句明白是什么意思即可。

5/16 变量的类型可以被强行改变

我们在使用变量的时候，其类型可以暂时被强行改变成另一个类型，方法是：（另一个类型）变量。例如：

```
int a = 5;
```

```
System.out.println((double)a);
```

`(double)a` 强行把 `int` 型的 `a` 暂时变成了 `double` 型的，也就是从不能有小数暂时变成了可以有小数，所以，此时输出的是 5.0 而不是 5。

同样，`double` 型的也可以暂时被强行转换成 `int` 型的，例如：

```
double b=5.9;
```

```
System.out.println((int)b);
```

(int)b 强行把 double 型的 b 暂时变成了 int 型的,也就是从可以有小数暂时变成了不能有小数,所以,此时输出的是 5 而不是 5.9,注意,此时小数部分会被直接舍弃,不会四舍五入。

考试的时候,有些小题会要求强行改变变量的类型,要记住改变的方法。

6/16 变量、类的命名原则

在前面的视频里,我们定义变量、类的时候,一般起的都是 a, b, c, a1, a2, A, B 这样的名字,名字基本上可以随便起,只要遵循以下四个原则就行:

① 不能使用 java 里有功能的单词;

【不能用 int、double、char、string、boolean、class、if、do、while、for、default、class、final、this、super、public 等作名字】

② 不能用数字作变量名开头;

【可以起名是 a1、a2,但是不可以起名是 1a、2a】

③ 不能包含除了_和\$之外的符号

【可以起名是 a_1、\$a,但是不可以起名是 a&1、a@1……】

(_ 很好理解,很常用很普遍嘛,至于 \$,没得说,java 设计师真掉钱眼里了……)

咋样,这就记住了吧 0(n_n)0

④ 不能在中间带空格

【可以起名是 a_1,但是不可以起名是 a 1】

考试的时候,这个点爱考选择。

7/16 定义数组的其他方法

在前面的视频里,我们讲过,要想同时定义很多个变量的时候,方法是用数组的方法,如:

```
int wudi[] = new int[10000];
```

【定义 10000 个 int 型变量,这 10000 个变量包含 wudi[0]、wudi[1]到 wudi[9999]】

```
double wudi[] = new double[500];
```

【定义 500 个 double 型变量,这 500 个变量包含 wudi[0]、wudi[1]到 wudi[499]】

首先,上面两种定义方法里,[]的位置可以移动到 变量名(wudi) 的前面。

然后,在考试的时候,我们也有可能遇见其他定义数组的方式,像下面这些:

```
int wudi[] = new int[]{6,7,8,9};
```

```
int wudi[] = {6,7,8,9};
```

【定义 wudi[0]=6、wudi[1]=7、wudi[2]=8、wudi[3]=9,他们都是 int 型变量】

```
double wudi[] = new double[]{6.0,7.0};
```

```
double wudi[] = {6.0,7.0};
```

【定义 wudi[0]=6.0、wudi[1]=7.0,他们都是 double 型变量】

考试的时候,如果看到这样的语句,知道是啥意思就行。

8/16 数组的 length

请记住 数组名.length = 数组里变量的个数。

如: int wudi[] = new int[10000];

则 wudi.length=10000

```
double wudi[] = new double[500];
```

则 wudi.length=500

考试的时候,有些程序里可能看到 哈哈.length,看到了认识即可。

9/16 二维数组

请理解下面语句的意思,并记忆对应知识点:

```
int[][] wudi = new int[2][3];
```

【定义了 2 行 int 型变量,每行 int 型变量包含 3 个变量, wudi.length=2】

【第 1 行变量有 wudi[0][0]、wudi[0][1]、wudi[0][2]共 3 个, wudi[0].length=3】

【第 2 行变量有 wudi[1][0]、wudi[1][1]、wudi[1][2]共 3 个, wudi[1].length=3】

```
int[][] wudi = new int[][]{{1,2,3},{4,5}};
```

```
int[][] wudi = {{1,2,3},{4,5}};
```

【定义了 2 行 int 型变量，`wudi.length=2`】

【第 1 行变量有 3 个变量 `wudi[0][0]=1`、`wudi[0][1]=2`、`wudi[0][2]=3`，
`wudi[0].length=3`】

【第 2 行变量有 2 个变量 `wudi[1][0]=4`、`wudi[1][1]=5`，`wudi[1].length=2`】

考试的时候会出 { 关于 length
数组名[?][?] = ? 这样的选择题。
第几行第几个变量是几

10/16 数组变量的特殊之处

请记住下面关于数组变量的特殊结论，小题爱考：

① 若已定义 一维数组 1，又定义 一维数组 2 = 一维数组 1；

那么，给 一维数组 1[相同数] 或 一维数组 2[相同数] 的其中一个赋值的时候，也会同时给另一个赋值；

比如说：已知 `int[] A=new int[100];`

`int[] B=A;`

那么此时，如果我们执行命令 `A[0]=1;`

就相当于同时执行了命令 `B[0]=1;`

如果我们执行命令 `B[1]=2;`

就相当于同时执行了命令 `A[1]=2;`

② 若已定义 二维数组，又定义 二维数组[数 1]=二维数组[数 2]；

那么，给 二维数组[数 1][相同数] 或 一维数组[数 2][相同数] 的其中一个赋值的时候，也会同时给另一个赋值；

比如说：已知 `int[][] A=new int[2][3];`

`A[0]=A[1];`

那么此时，如果我们执行命令 `A[0][0]=1;`

就相当于同时执行了命令 `A[1][0]=1;`

如果我们执行命令 `A[0][1]=2;`

就相当于同时执行了命令 `A[1][1]=2;`

小题爱考上面这两点

11/16 简写的计算语句

请理解下面语句的意思，记住规律：

$c += a * 5;$ 意思是 $c = c + (a * 5);$

$d *= a + b;$ 意思是 $d = d * (a + b);$

$e /= a - 4;$ 意思是 $e = e / (a - 4);$

考试时遇见，认识即可。

12/16 带 ++ 或者 -- 的语句

请理解下面语句的意思，记住规律：

++ 或者 -- 在变量前：先让变量自加 1 或者自减 1 再执行语句

【++ 与 -- 之间不能有空格】

$c = ++a * b;$ 意思是：先让 $a = a + 1$ ，再让 $c = a * b$

$c = a - --b;$ 意思是：先让 $b = b - 1$ ，再让 $c = a - b$

++ 或者 -- 在变量后，先执行语句，再让变量自加 1 或者自减 1

【++ 与 -- 之间不能有空格】

$c = a ++ * b;$ 意思是：先让 $c = a * b$ ，再让 $a = a + 1$

$c = a -- - b;$ 意思是：先让 $c = a - b$ ，再让 $a = a - 1$

考试时遇见，认识即可。

13/16 花里胡哨的输出语句

在前面的视频中，我们讲过这种输出语句：

`System.out.println("文字" + 变量或式子);`

执行后，会输出：文字变量或式子的结果

考试的时候，可能有的小题会考的比较杂，比如可能会出现这种情况：

```
System.out.println(变量或式子 1+ 变量或式子 2+ “文字”+变量或式子 3+变量或式子 4);
```

此时，请记住，文字前的内容，统一当做是一个式子，文字后面的内容，一个+隔出一个式子。

所以，上面的语句其实相当于是

```
System.out.println(变量或式子 1+ 变量或式子 2+ “文字”+变量或式子 3+变量或式子 4);
```

执行后，会输出

(变量或式子 1+ 变量或式子 2)的结果文字变量或式子 3 的结果变量或式子 4 的结果

比如说 `System.out.println(1+2+ “厉害”+ 3+ 4);`

执行后，会输出 3 厉害 34

考试时，可能会考选择或者填空。

此外，输出语句除了 `System.out.println(“文字”+变量或式子);`

还可以写成 `System.out.print(“文字”+变量或式子);`

他俩的区别是，用 `println` 输出之后，会换一下行，

这样，可以避免所有输出的东西都堆积在一起，后面如果再输出东西，会在下一行输出，层次分明。

而 `print` 没有换行功能，如果用 `print` 输出之后，后面再输出东西，会在这次输出结果的后面输出。

例如：

```
System.out.println(“猴博士” );
```

```
System.out.println(“大天才” );
```

输出的结果是：

猴博士

大天才

```
System.out.print(“猴博士” );
```

```
System.out.print(“大天才” );
```


输出的结果是：

猴博士大天才

考试时，可能考判断题。

14/16 二选一语句

请记住 条件 ? 式子 1 : 式子 2 在 $\begin{cases} \text{条件成立时} & , \text{整体} = \text{式子 1} \\ \text{条件不成立时} & , \text{整体} = \text{式子 2} \end{cases}$

比如：c = a > b ? a : b ;

那么，意思是，如果当前情况下 a > b ，那么 c=a，

否则，

c=b

考试时，可能会考选择或者填空。

15/16 switch 语句

请记住下列语句及规律。

```
switch (某变量) {
```

```
case 值 1:
```

代码 1

```
case 值 2:
```

代码 2

```
case 值 3:
```

代码 3

```
.....
```

```
case 值 n:
```

代码 n

```
default:
```

代码其它

```
}
```

某变量 = 值 1 时，以上语句会执行代码 1、代码 2、……、代码 n、代码其它；

某变量 = 值 2 时，以上语句会执行代码 2、代码 3、……、代码 n、代码其它；

某变量 = 值 3 时，以上语句会执行代码 3、代码 4、……、代码 n、代码其它；

……

某变量 = 值 n 时，以上语句会执行代码 n、代码其它；

某变量 不等于 值 1、值 2、……、值 n 时，以上语句只执行代码其它。

【注意，执行代码时，若执行到 break，则 break 后面的代码语句都不用执行】

考试时可能考小题

16/16 让程序中某些部分不起作用

java 中规定，在程序中，

① 每一行中，“//” 开始一直到该行末尾的部分

② “/*” 和 “*/” 以及它们之间的部分




上面两个部分的内容不起任何作用，会被自动忽略，相当于没写：

例如：

```
public class Test{  
    public static void main(String args[]){  
        // int a=10;  
        int b=4; // 一些文字说明  
        /* a = a / 2;  
        b = b * a; */  
    }  
}
```

相当于

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

```
int b=4;   
  
  
}  
}
```

考试的时候，看到语句中包含这两种格式的符号，要知道哪一部分没起作用。