Java学习笔记

要学Java了，作为最难上手的编程语言之一，我要先偷渡一点
首先先在这里写几个常用命令和小提示：
javac xxx.java //编译程序为class
javap xxx //反编译程序
Java的注释分单行和多行注释：
单行注释是// 就像python的#
多行注释是/* */ 就像python的''' '''

简单认识

这部分内容可以说是一个新手教程笔记，我对编程风格没有偏好，能出成果就行

简单语法差异

文本输出以及严谨性

```
public class Hello {
    public static void main(String args[]) {
        System.out.println("大家好啊！");
    }
}
```

和

```
public class App {
    public static void main(String[] args) throws Exception {
        System.out.println("Hello, World!");
    }
}
```

都能正常输出文本，而两者对应的class（类）不同，所以在编译后对应的程序名不同，一个是Hello，一个是App
第二行的String[] args和String args[]都正确，但是String[] args更符合规范，第二个的throws Exception是在下面的println出错时报错，这样可以更严谨，虽然在这里没什么作用

入口方法解释

入口方法对于Java程序代码是必不可少的，这是Java程序的执行入口，每一个修饰词都有自己的功能与含义，我会随着接触内容而扩充该部分内容的

例子

第一个以常见的public static void main(String[] args)为例：

`public`

- 访问修饰符，表示这个方法是 **公共的**，任何地方都能调用
- JVM（Java 虚拟机）在启动程序时需要访问这个方法，所以必须是 `public`

`static`

- 静态方法，属于类本身，而不是某个对象
- JVM 在启动时还没有创建对象，所以 `main` 必须是 `static`，这样可以直接通过类调用

`void`

- 返回值类型，表示这个方法 **没有返回值**
- 程序入口只需要执行，不需要返回结果给 JVM

`main`

- 方法名，固定写法
- JVM 会自动寻找这个名字作为程序入口

`String[] args`

- 方法参数，表示一个 **字符串数组**
- 用来接收命令行传入的参数

简单程序解释

面向对象结构示例

```
public class Fatguy {
    int height;
    int weight;
    int waistline;
    String gut;
    void speak(String say) {
        System.out.println(say);
    }
}
class A {
    public static void main(String[] args) {
        Fatguy guy;
        guy = new Fatguy();
        guy.height = 180;
        guy.weight = 150;
        guy.waistline = 130;
        guy.gut = "一个大胖子";
        System.out.println("身高: " + guy.height + "cm");
        System.out.println("体重: " + guy.weight + "kg");
        System.out.println("腰围: " + guy.waistline + "cm");
        System.out.println(guy.gut);
        guy.speak("终究败给良子");
    }
}
```

程序输出：
身高：180cm
体重：150kg
腰围：130cm
一个大胖子
终究败给良子

这个程序定义了一个类（Fatguy）和一个主类（A），在main方法里，声明了一个`Fatguy`类型的变量`guy`，使用`new Fatguy()`创建对象并赋值给`guy`，给对象的属性赋值（height、weight、waistline、gut）并打印这些属性
调用对象的方法`speak()`也是被定义出来的，其目的就是让下面的主类被调用使用，因为括号内的是String，所以下面的调用插入内容也得是字符串
然后到主类A这里，public static void main(String[] args)是程序的入口方法，这个方法的作用是告诉JVM：这是一个**公共的静态方法**，没有返回值，名字叫`main`，参数是一个字符串数组

编译提示：这个文件的名字必须和类一致（VSCode和IDEA会自动修正文件名），然后在相同文件夹内执行javac Fatguy.java，会得到两个文件（Fatguy.class和A.class），执行java A即可执行主程序，两个文件必须在同一位置，而且不能执行Fatguy.class，否则会报错：
在类 Fatguy 中找不到 main 方法, 请将 main 方法定义为:public static void main(String[] args)，否则 JavaFX 应用程序类必须扩展javafx.application.Application
这倒提醒我了，就好像Python可以用同一文件夹的另一个Py def程序作为插件使用，而def程序不能独立使用

基本数据类型处理

Java有标识符和关键字这两个定义
标识符是开发者**自己定义的名字**，但**不能和关键字和true, false, null相同**，标识符的**第一个字符不可以是数字**，可以包含字母，下划线，美元符号，数字，而且需要区分大小写（hello和Hello是不同的标识符）
关键字是有特定用途的且被赋予一定意义的单词，它们是本身就存在的，后续会接触不同的关键字

基本数据类型

Java语言有8中基本数据类型，分类后是：

- 逻辑类型：boolean
- 整数类型：byte, short, int, long
- 字符类型：char
- 浮点类型： float, double

逻辑类型

boolean的常量只有两个：true和false，boolean可以声明逻辑变量，声明时可以赋初值，例　：

```
boolean fat = true, on = true, off = false, isThin;
```

这是通过boolean给fat, on, off, isThin赋值，　果在类里定义没有赋值（　后面的isThin，则默认赋值false），在方法定义则必须赋值：

```
public class Test {
    boolean isThin; // 成员变量，默认值 false

    public static void main(String[] args) { //这是方法定义
        boolean flag; // 局部变量
        // System.out.println(flag); // ❌ 编译错误：未初始化

        Test t = new Test(); //这是类定义
        System.out.println(t.isThin); // ✅ 输出 false
    }
}
```

整数类型

整数类型分四种：
1. int型
int型支持常量（二进制，八进制，十进制，十六进制）
赋值方式和boolean基本一致，十六进制定为0x12ab、0X12ab或x12ab都可以，八进制需要以数字0作前缀0123，二进制可以定为0b11或0B11，但不能赋空值

```
int h=180,w=0x9b,l=0230,e=0b1;
```

2. byte型
并没有byte表示常量的表示法，但是一定范围的int型可以转换为byte型变量
byte型变量分配一字节内存占八位，所以byte型变量的取值范围是-2^7~2^7-1（-128~127 共256个整数）
因为不管是什么整数值没有强调都会被认为是int，　果要将其作为byte类型运算，可以这样做：(byte)-11, (byte)20; //把-11和20转换为byte类型

3. short型
和byte型差不多原理，唯一区别是short型分配2字节内存，占16位，所以其取值范围是-2^{15}~2^{15}-1 (-32768-32767共65536个整数）

4. long型
和上面两个差不多原理，一个区别是long型分配8字节内存，占64位，所以其取值范围是-2^{63}~2^{63}-1（中间共18446744073709551616个整数），另一个区别就是long型常量用后缀L表示，　：
123L（适用于二进制，十进制，八进制，十六进制的数字）

字符类型

字符char的常量就是任意字符，包括Unicode字符，单个字符常量只能包含一种字符

```
char ch = 'Abcd', num = '1234', home = '房子', alt0147 = 'Ň';
```

char型变量分配2字节内存，占16位，最高位不是符号位，没有负数的char，和short型整数差不多，取值范围是0~65535

在Java中，a对应的字符码是97，所以：

```
public class charset {
    public static void main(String[] args) {
        char x = 97;
        System.out.println(x);
    }
}
```

等同于：

```
public class charset {
    public static void main(String[] args) {
        char x = 'a';
        System.out.println(x);
    }
}
```

果字符存在冲突，不能直接输入使用，可以使用以下转义符常量：

```
\n //换行
\b //退格
\t //水平制表
\' //单引号
\" //双引号
\\ //反斜杠 等
```

比 ：

```
public class charset {
    public static void main(String[] args) {
        char x = "老子喜欢双引号"";
        System.out.println(x);
    }
}
```

要改为：

```
public class charset {
    public static void main(String[] args) {
        String x = "我喜欢双引号\"";
        System.out.println(x);
    }
}
```

Java中，可以用字符在Unicode表中的排序位置的十六进制转义，格式是'\u****'，例 ：'\u0041'是字符A，'\u0061'是字符a （十六进制的61就是十进制的97）

字符查询

可以使用类型转换显示一些字符在Unicode表中的位置：

```
public class chars {
    public static void main(String[] args) {
        char word = '好';
        int numb = 20320;
        System.out.println(word + " 在Unicode上的位置是: " + (int)word);
        System.out.println(numb + " 对应的字符是: " + (char)numb);
        numb = 114514;
        System.out.println(numb + " 对应的字符是: " + (char)numb);
    }
}
```

这个代码涉及到另一个知识点：numb在被修改前，首先要定义其类型

浮点类型

浮点型分为float（单精度）型和double（双精度）型
1. float
float常量可以表示为：
123.456f/F（小数表示法）
2e11f/F（指数表示法）（2乘10的11次方）而且输入整数会自动为输出内容加.0
使用float变量来声明，float变量只存储8位有效数字，比　声明12345.6789为float后输出值12345.679
float型变量分配4字节内存，占32位

2. double
double常量的表示方式和float差不多，后缀d/D可加可不加：
123.456789，145.222d
double保留16位有效数字，一个变量分配8字节内存，占64位

类型转换运算

把一种基本数据类型变量赋给另一种变量涉及数据转换，下面这些类型涉及数据转换：

```
byte short char int long float double
//精度从低到高排列
```

低精度到高精度的转换可以自动进行：

```
int x = 100; //下面用int转换为float
float y;
y = x //输出100.0
```

高精度到低精度的转换需要使用类型转换运算：

```
int x = 91; //下面用int转换为char
char y;
y = (char)x; //输出[
```

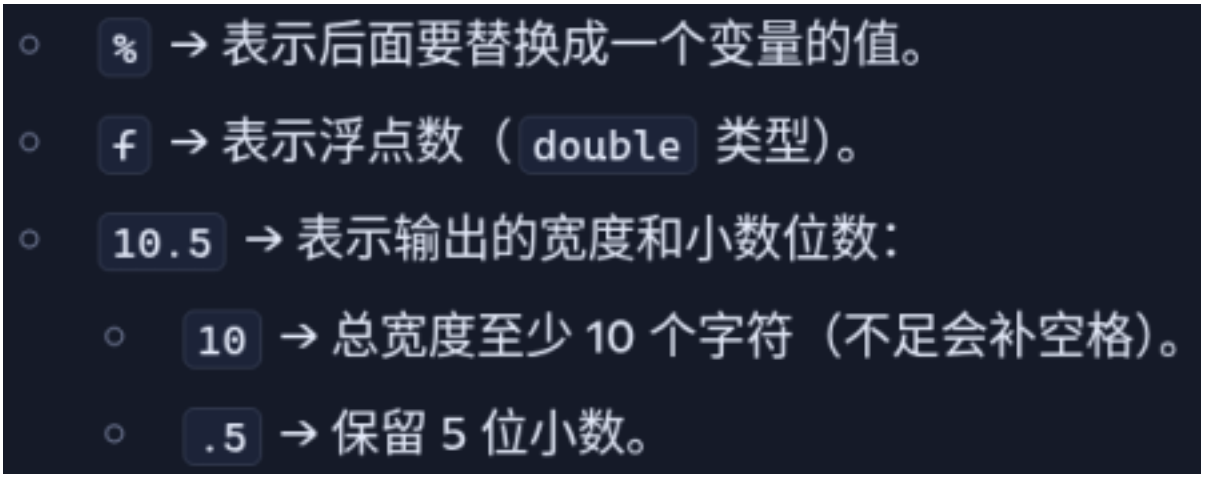
高精度到低精度的转换可能会导致精度损失

数据的输入输出

输入：Scanner是一个类，可以读取在命令行中输入的内容，下面是一个输入数字实现加法运算的例子：

```
import java.util.Scanner;
public class io {
    public static void main(String[] args) {
        System.out.println("用空格/回车分隔，输入几个数字\n" +
            "最后输入0，再按下回车结束"); //println支持换行输出
        Scanner x = new Scanner(System.in);
        double sum = 0; //先定义一个值为0的数据，数据加在它身上
        double y = x.nextDouble(); //开始捕捉数据
        while (y≠ 0) { //这也是为什么输入0后就会停止
            sum = sum + y;
            y = x.nextDouble();
        }
        System.out.printf("sum = % 10.5f\n", sum);
    }
}
```

其中"sum = % 10.5f\n"里各个字符的含义是：



输出：关于System.out.println()和System.out.print()，二者的区别就是，println输出数据后换行，换行例可参照上面的代码，编写代码时想换行写法也需要明确：

```
System.out.println("你好，
  这是不对的")
```

正确的写法是：

```
System.out.println("你好，" +
    "这个是对的")
```