link: null
title: 珠峰架构师成长计划
description: null
keywords: null
author: null
date: null
publisher: 珠峰架构师成长计划
stats: paragraph=82 sentences=130, words=1001

1. 安装配置 JDK

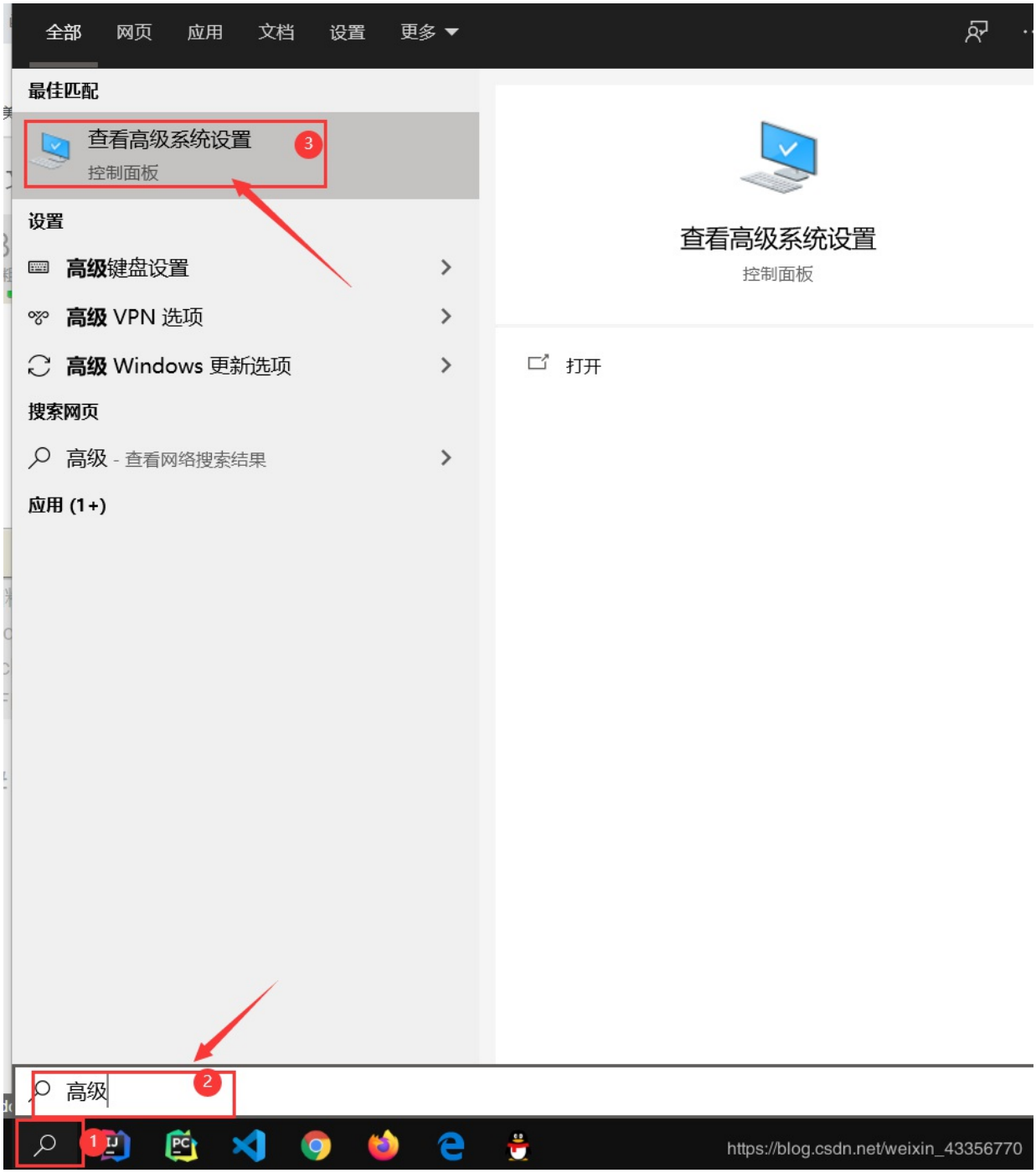
1.1 下载 JDK

- javase-jdk8-downloads (https://www.oracle.com/java/technologies/javase/javase-jdk8-downloads.html)
- jdk-8u251-windows-x64 链接：https://pan.baidu.com/s/1P7hUHQLpunhJ0yfUd85Bxw (https://pan.baidu.com/s/1P7hUHQLpunhJ0yfUd85Bxw) 提取码：aeuy

1.2 安装 JDK

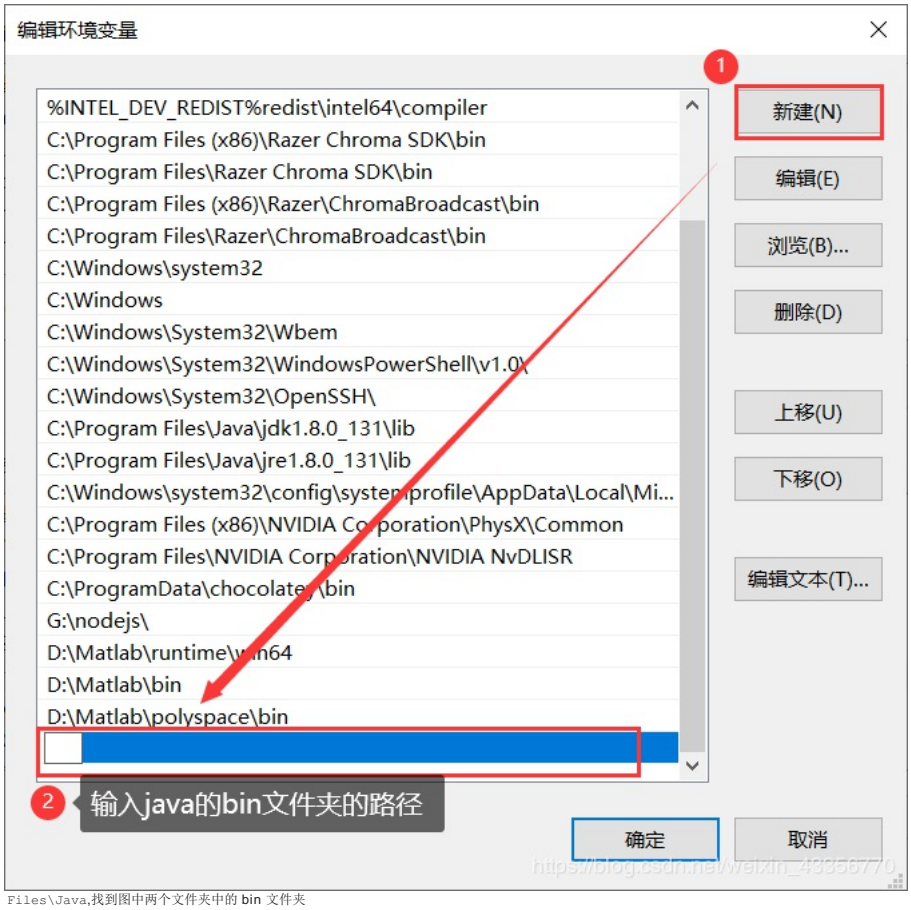
1.3 配置环境变量 #

- 左下角搜索（或者快捷键 WIN+Q ）==> 输入 `&#x9AD8;&#x7EA7;` ==> 查看高级系统设置

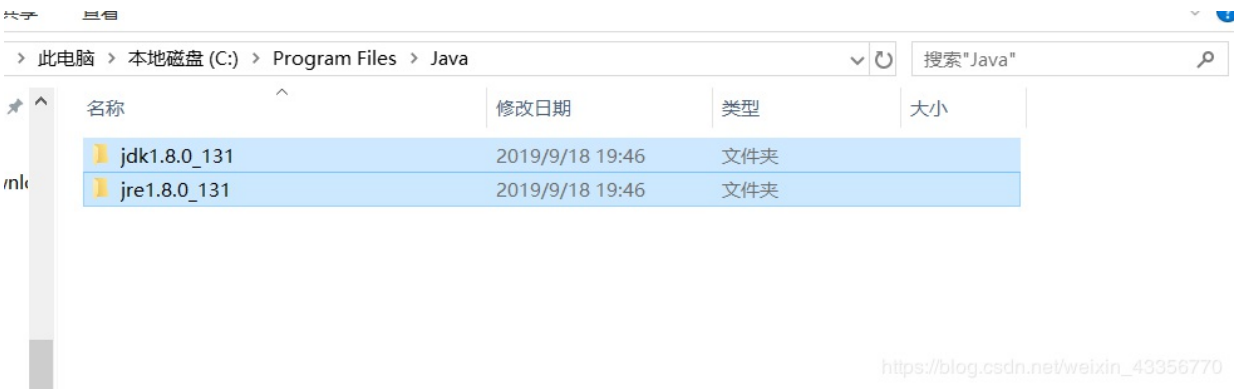

全部
网页
应用
文档
设置
更多
最佳匹配
查看高级系统设置
控制面板
3
设置
高级键盘设置
高级 VPN 选项
高级 Windows 更新选项
搜索网页
高级 - 查看网络搜索结果
应用 (1+)
查看高级系统设置
控制面板
打开
高级
2
1
https://blog.csdn.net/weixin_43356770

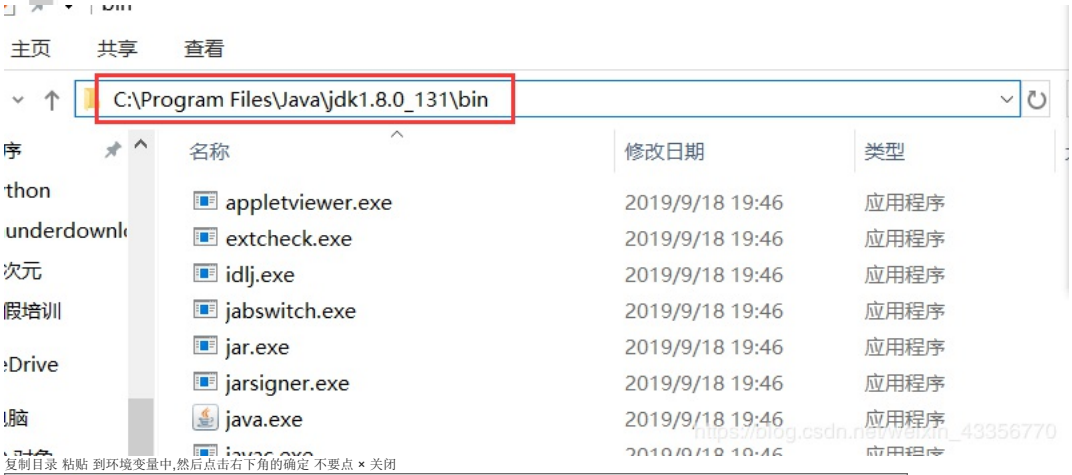
- 选择 高级 ——>环境变量

- 选择系统变量下的 Path（双击）

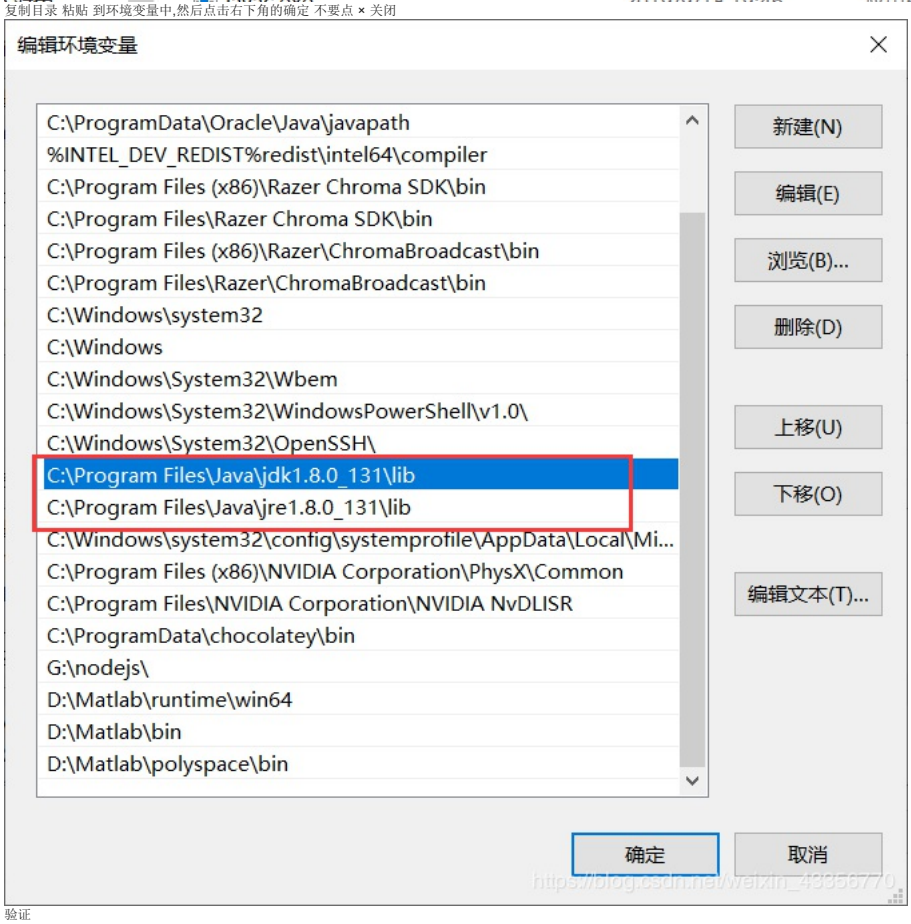


- 新建——>粘贴 java 下 jdk 和 jre 的 bin 文件夹的路径
- 如果是默认安装目录,则进入到这个目录, `C:\Program Files\Java`,找到图中两个文件夹中的 bin 文件夹

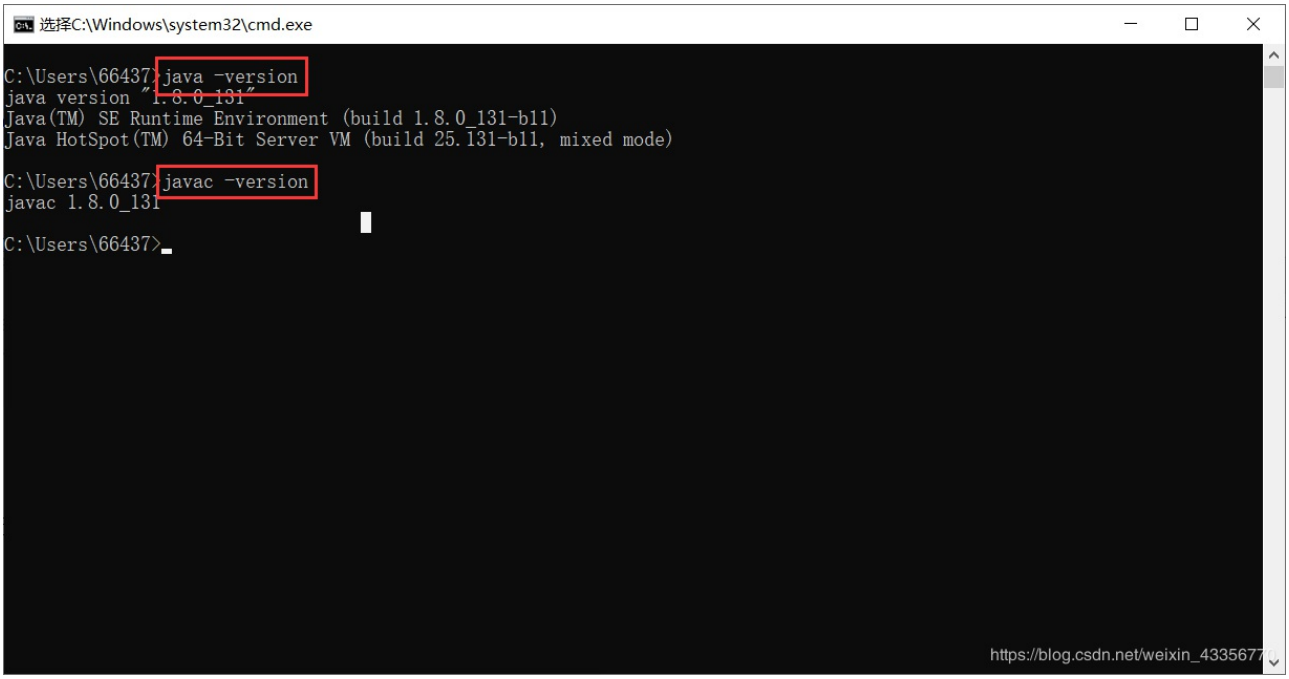




- 复制目录 粘贴 到环境变量中,然后点击右下角的确定 不要点 × 关闭



- 验证
- win+R 输入 cmd 运行 `java -version`



2. 开发步骤

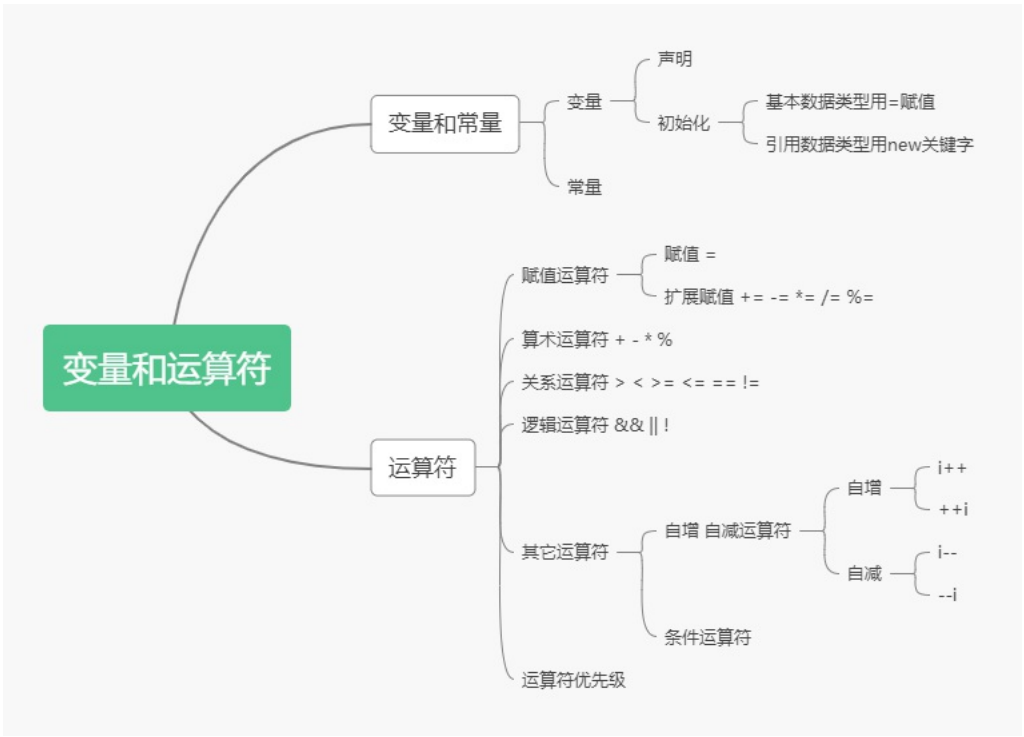
- 使用记事本编辑 .java为后缀的文件
- 使用 javac 命令编译 .java文件,生成 .class文件
- 使用 java 命令运行 .class 文件,得到程序结果
- 也可以直接安装 Java Extension Pack插件

```
public class HelloWorld {

    public static void main(String[] args) {

        System.out.println("hello");
    }
}
```

注意事项

- main 作为方法的入口, void 不可少
- Java 对大小写敏感
- 每一条 Java 语句以分号结束
- 不要漏写引号,要使用双引号

3. 变量\数据类型\运算符



3.1 数据类型

数据类型 说明 char(字符型) 用于存储单个字符,如：性别 男 女

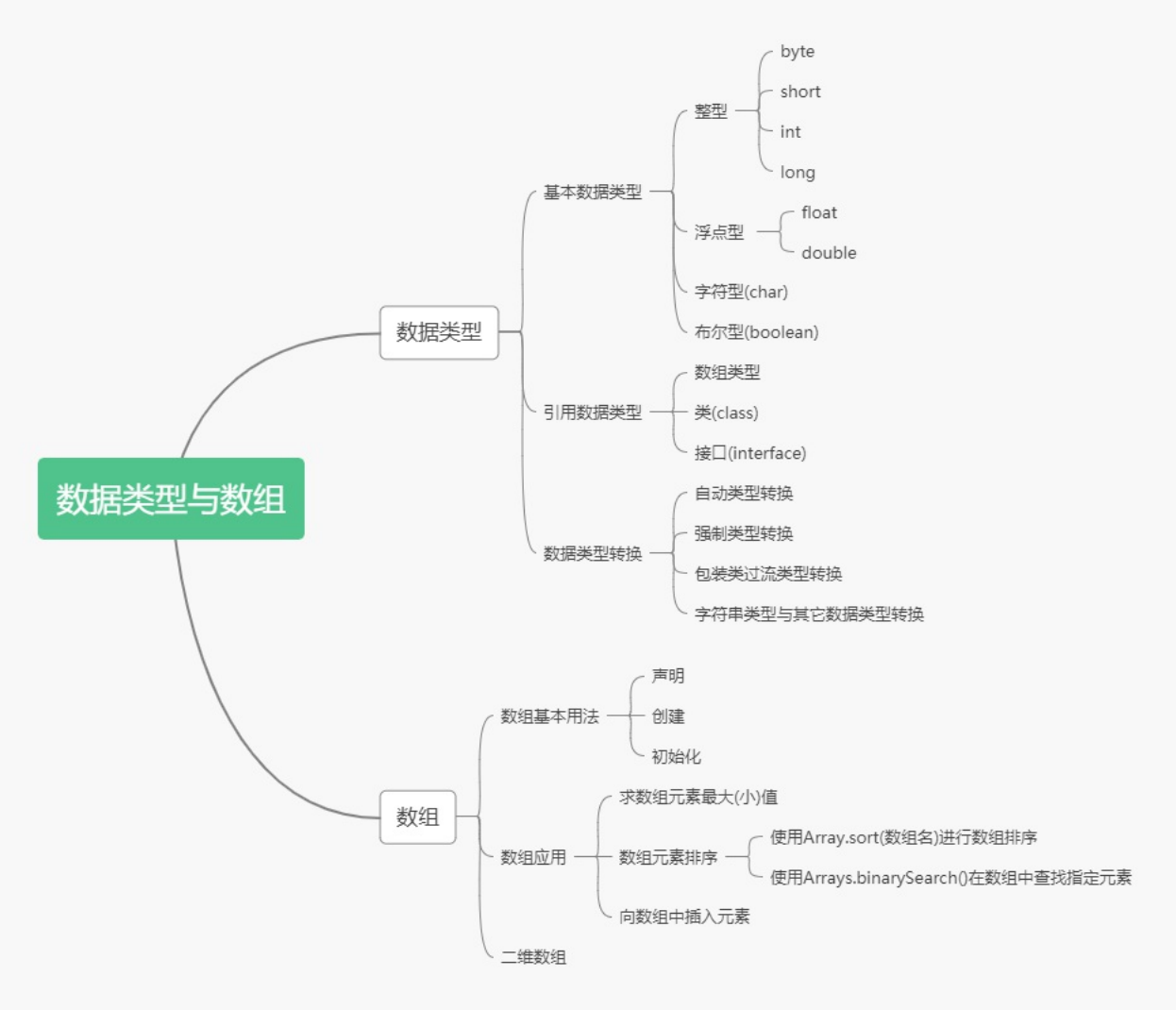
int(整型) 用于存储整数,如：一天的时间是 24 小时 double(双精度) 用于存储小数 如：早餐的价格 3.5 元 String(字符串) 用于存储一串字符,如 "我爱前端"

```
public class DataType {
    public static void main(String[] args) {
        char c = '珠';
        System.out.println(c);
        int d = 3;
        double f = 4.4;
        String g = "峰";
    }
}
```

3.2 运算符

```
import java.util.Scanner;

public class Sum {
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
        System.out.println("please input first number");
        int num1 = input.nextInt();
        System.out.println("please input second number");
        int num2 = input.nextInt();
        int result = num1 + num2;
        System.out.println("result is " + result);
        input.close();
    }
}
```

4. 数组



4.1 什么是数组

- 数组是一个变量,存储相同数据类型的一组数据
- 声明一个变量就是在内存空间划出一块合适的空间
- 声明一个数组就是在内存中划出一串连续的空间

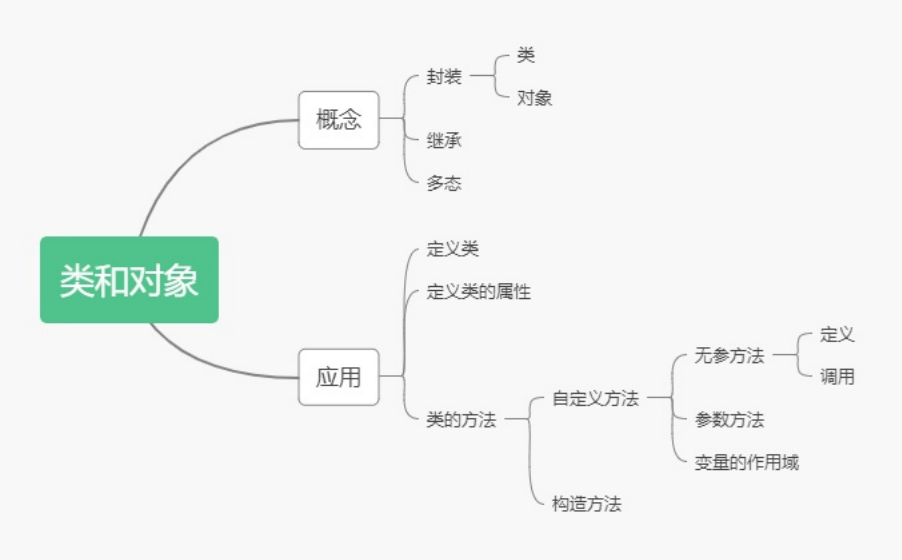
4.2 数组基本要素

- 标识符：数组的名称，用于区分不同的数组
- 数组元素：向数组中存放的数据
- 元素下标：对数组元素进行编号，从 0 开始，数组中的每个元素都可以通过下标来访问
- 元素类型：数组元素的数据类型

- 注意事项
    - 数组长度固定不变，避免数组越界
    - 数组中的所有元素必须属于相同的数据类型

```
public class Array {
    public static void main(String[] args) {
        int[] arr;
        arr = new int[3];
        arr[0] = 1;
        arr[1] = 2;
        arr[2] = arr[0] + arr[1];
        System.out.println(arr[2]);

        int[] arr2 = new int[] { 1, 2, 3 };
        System.out.println(arr2);
    }
}
```

5. 类和对象



- 类是抽象的概念,是一个模板
- 对象是一个你能看的到摸的到的具体实体
- 类的方法定义类的某种行为或功能

5.1 方法定义

- 调用带参数方法 对象名.方法名(参数 1,参数 2,参数 3....参数 n)
- 实参的类型 数量 顺序都要与形参一一对应
- 形参和实参数据类型不一致会报错
- 注意事项
    - 如果方法的返回值类型为 `void`,方法中不能有 `return`返回值
    - 一个方法不能有多个返回值
    - 多个方法不能相互嵌套定义
    - 不能在方法外部直接写程序逻辑代码

```
  (){

}
```

5.2 类

```
class Person {
    String name;
    int age;

    public String getName() {
        return this.name;
    }
   public String getName(String prefix) {
        return prefix+this.name;
    }
    public void setName(String name) {
        this.name = name;
        int score;
        System.out.println(getName());
        System.out.println(age);

    }
}

class Student extends Person {

}

public class Entry {
    public static void main(String[] args) {
        Person p = new Person();
        p.setName("zhufeng");
        System.out.println(p.getName());

        Student s = new Student();
        s.setName("jiagou");
        System.out.println(s.getName());
    }
}
```

6. 字符串

- 可以使用 String 对象存储字符串
- String 类位于 java.lang 包中,具有丰富的方法

```
public class MyString {
    public static void main(String[] args) {
        String s1 = "Hello";
        String s2 = new String();
        String s3 = new String("world");
        System.out.println(s1.length());
        System.out.println(s1.equals(s3));
        System.out.println(s1 == s3);

        StringBuffer sb = new StringBuffer();
        sb.append("hello");
        sb.append("world");
        System.out.println(sb.toString());
    }
}
```

7. 类

7.1 构造函数

- 无返回值类型,名称与类名相同,可以指定参数
- 构造方法可以重载
- static 静态方法用类名调用
- final 修饰的变量称为常量,值固定不变

```
class Animal {
    void talk() {
        System.out.println("Animal");
    }
}

final class Dog extends Animal {
    void talk() {
        System.out.println("Dog");
    }
}

class Golden extends Dog {
    void talk() {
        System.out.println("Dog");
    }
}

class Cat extends Animal {
    void talk() {
        System.out.println("Cat");
    }
}

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

    }
}
```

7.2 访问修饰符

访问修饰符 本类 同名 子类 其它 private √ X X X 默认 √ √ X X protected √ √ √ X public √ √ √ √

7.3 封装 继承 多态

7.3.1 封装

7.3.2 继承

- 符合 is a的关系设计时使用继承,使用 extends关键字
- 继承是代码重用的一种方式
- final 类不能被其它类继承
- super 关键字用于访问父类成员
    - super 只能出现在子类的方法和构造方法中
    - super 访问构造方法时,只能是第一句
    - super 不能访问父类的 private 成员
- 重写
    - 方法名相同
    - 参数列表相同
    - 返回值类型相同或者是其子类
    - 访问权限不能严于父类
- 抽象类和抽象方法
    - 抽象类不能被实例化
    - 可以有 0 或多个抽象方法
    - 非抽象子类必须 重写父类所有的抽象方法

7.3.3 多态

- 同一个引用类型，使用不同的实例而执行不同操作
- 子类重写父类的方法,运行时使用父类的类型

```
class Animal {
    void talk() {
        System.out.println("Animal");
    }
}

final class Dog extends Animal {
    void talk() {
        System.out.println("Dog");
    }
}

class Cat extends Animal {
    void talk() {
        System.out.println("Cat");
    }
}
```

8. 集合框架

- Java 集合框架提供了一套性能优良、使用方便的接口和类，它们位于 java.util 包中
- Collections 提供了对集合进行排序、遍历等多种算法实现
- Collection 接口存储一组不唯一，无序的对象
    - List 接口存储一组不唯一，有序（插入顺序）的对象
        - ArrayList ArrayList 实现了长度可变的数组，在内存中分配连续的空间。遍历元素和随机访问元素的效率比较高
            - add(Object o)
            - add(int index,Object o)
            - int size()
            - get(int index)
            - contains(Object o)
            - remove(Object o)
            - remove(index index)
        - LinkedList LinkedList 采用链表存储方式。插入、删除元素时效率比较高
            - addFirst(Object o)
            - addLast(Object o)
            - getFirst()
            - getLast()
            - removeFirst()
            - removeLast()
    - Map Map 接口存储一组键值对象，提供 key 到 value 的映射
        - HashMap
            - put(Object key,Object value)
            - get(Object key)
            - remove(Object key)
            - int size()
            - Set keySet()
            - Collection values()
            - boolean containsKey(Object key)
        - TreeMap
    - Set 接口存储一组唯一，无序的对象
        - HashSet
        - TreeSet

```java
import java.util.ArrayList;
import java.util.List;

public class MyArrayList {
    public static void main(String[] args) {
        List list = new ArrayList();
        list.add(1);
        list.add(2);
        list.add(3);
        System.out.println(list.size());

        Map map = new HashMap();
        map.put("A","优秀");
        System.out.println(map.get("A"));

        Set set = new HashSet();
        set.add("A");
        set.add("B");
        System.out.println(set.size());
    }
}
```