组长:魏志航

组员:兰天阳，彭尹垚，张世纪，叶翔昊，谢泽川

总体情况：

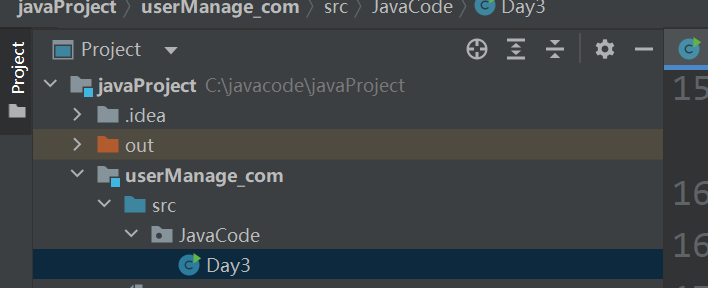
今天所有组员仍100%完成任务。

**魏志航**

今天完成的任务主要包括：

1.安装java的高级集成开发工具idea

2.在idea中按照project->module->package->class的顺序创建新项目



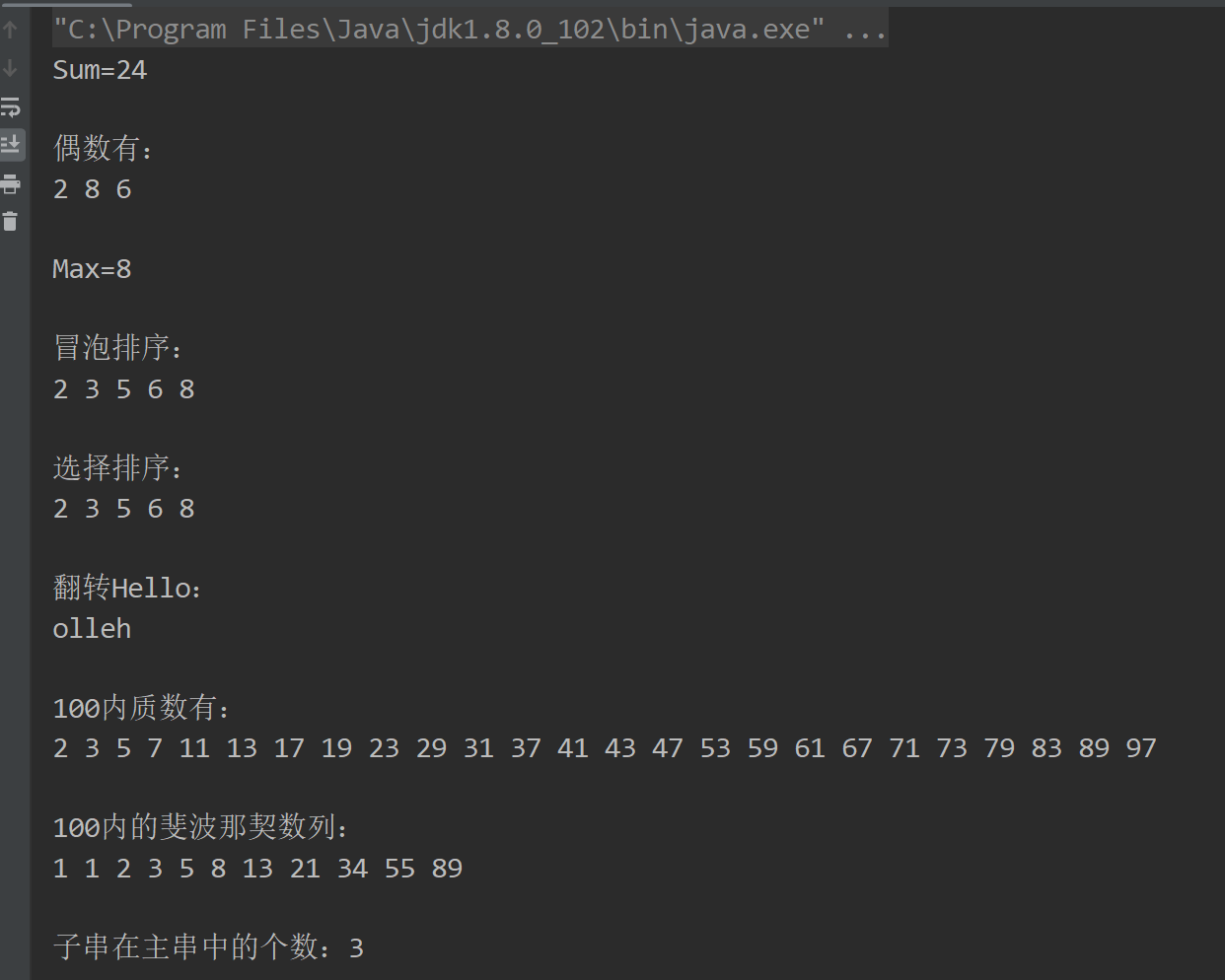
3.idea中的psvm和sout的快捷方式的使用

4.数组的创建和赋值

5.面向对象，包括创建类（包含属性和方法）、静态方法

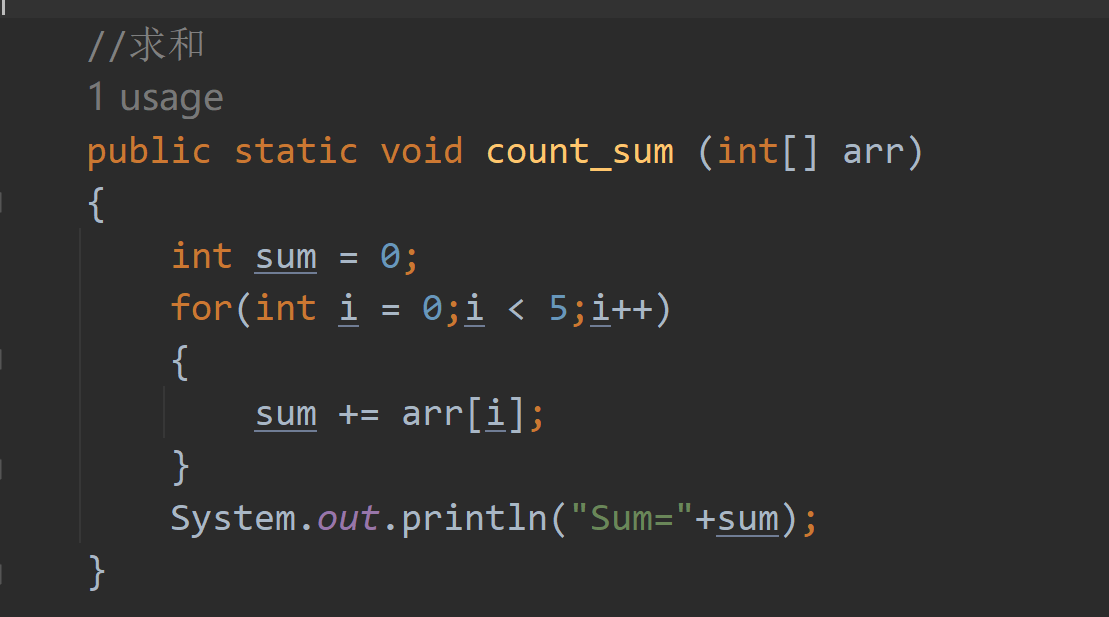
上述知识的综合运用融汇在了8个编程题目中：

首先是8道题的代码的运行结果：

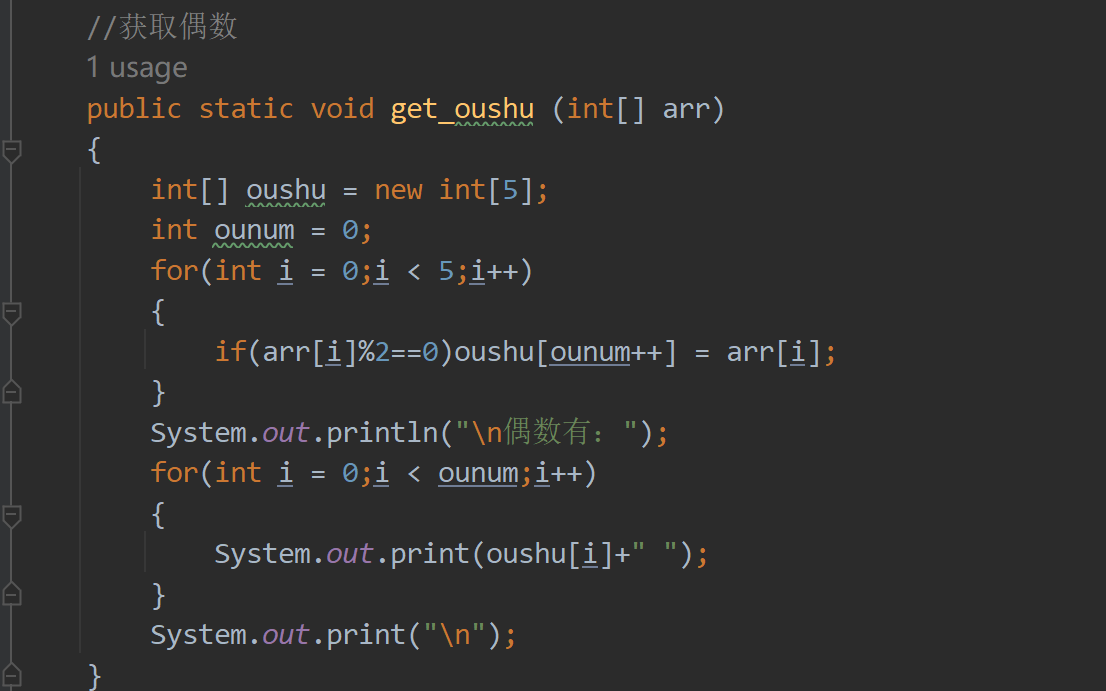


每题封装的静态方法的代码：

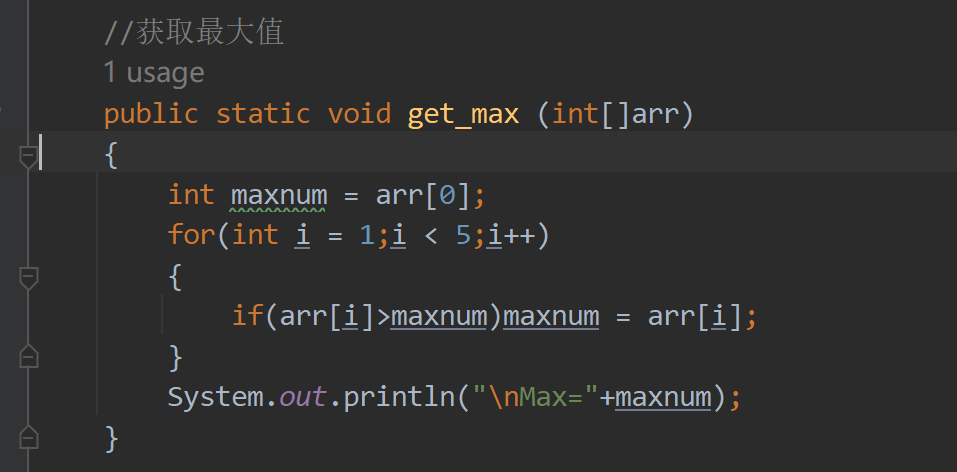
1.



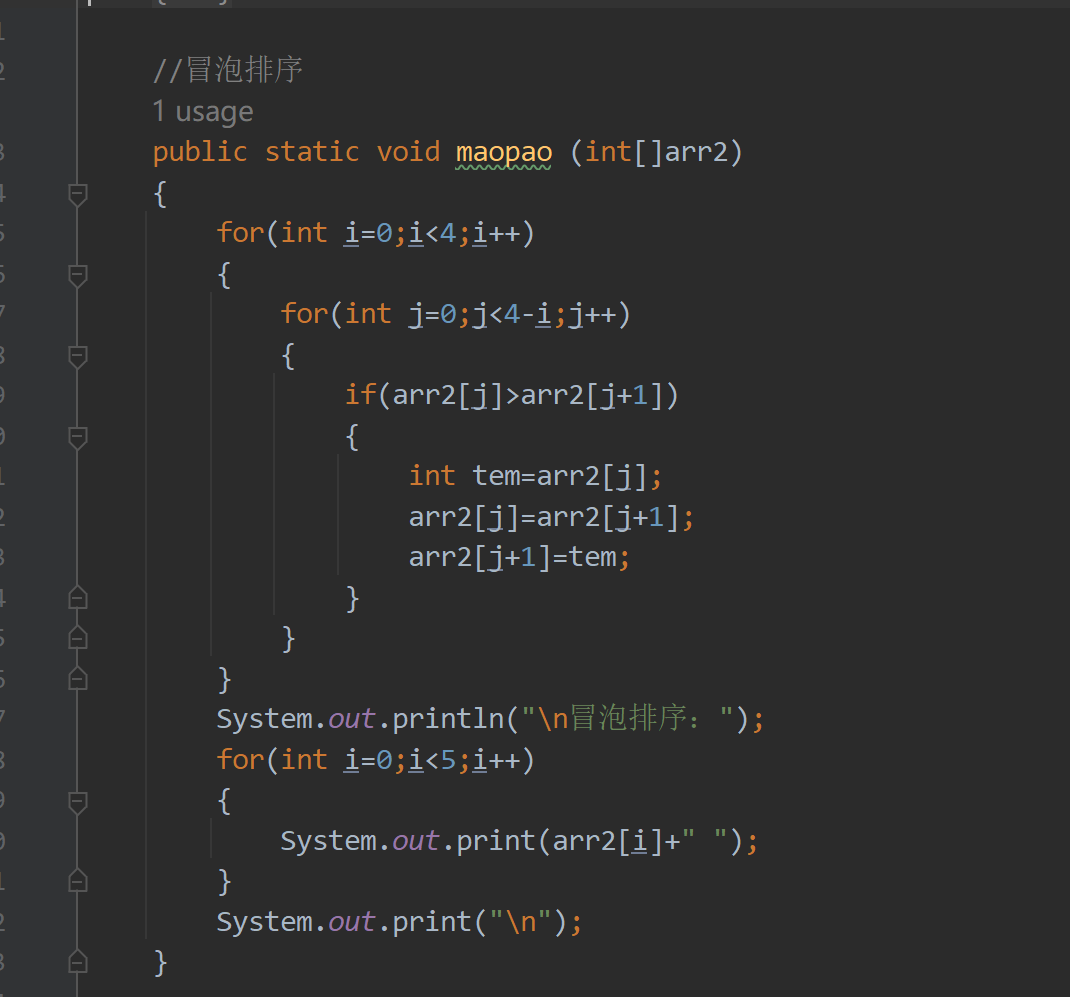
2.

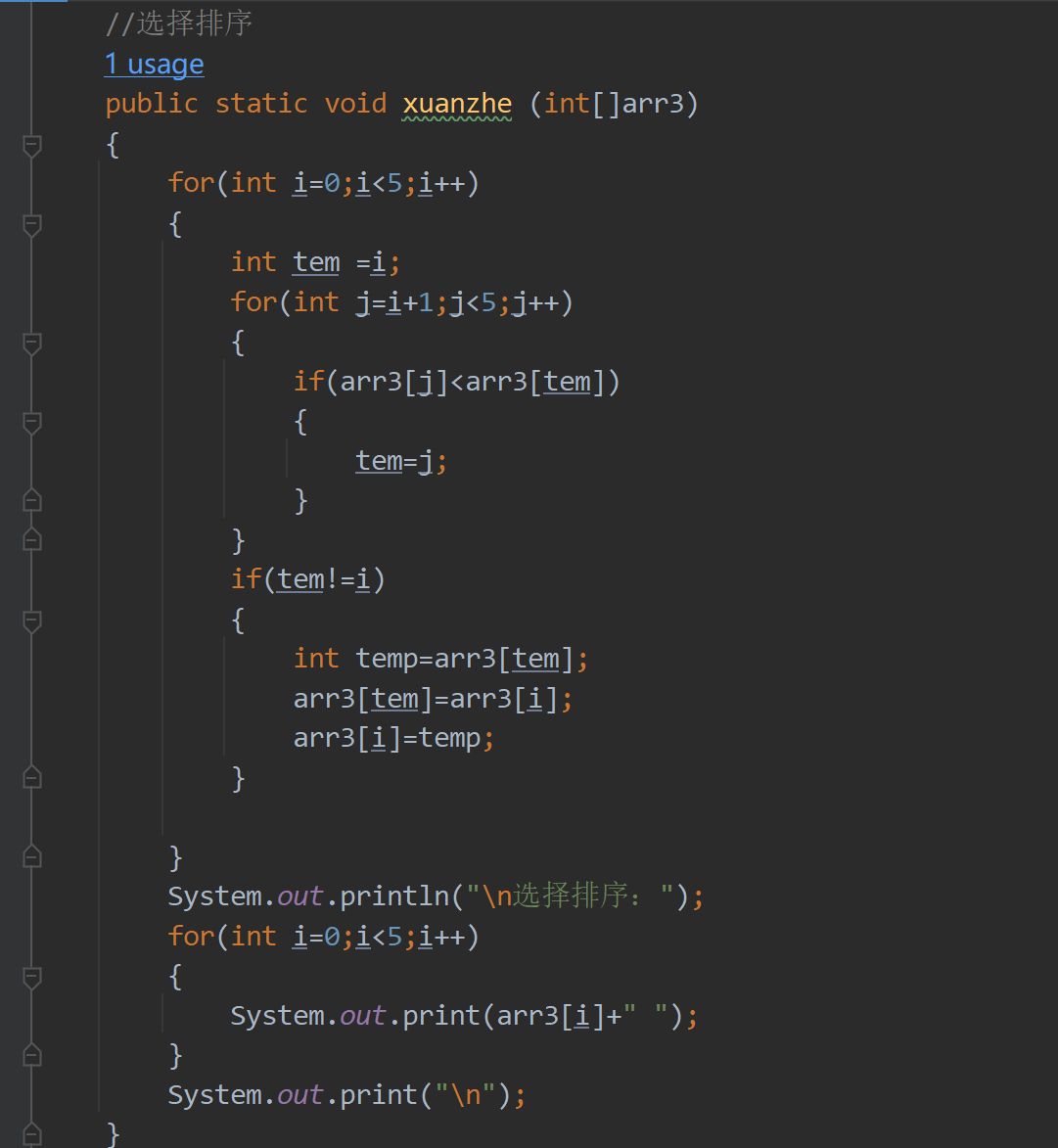


3.

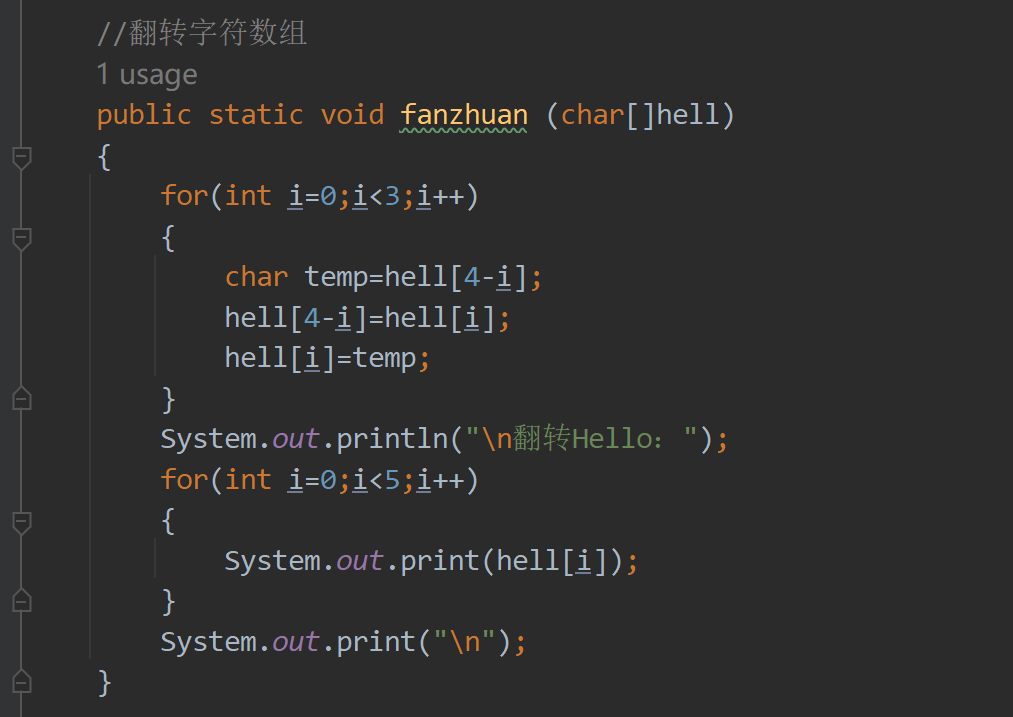


4.

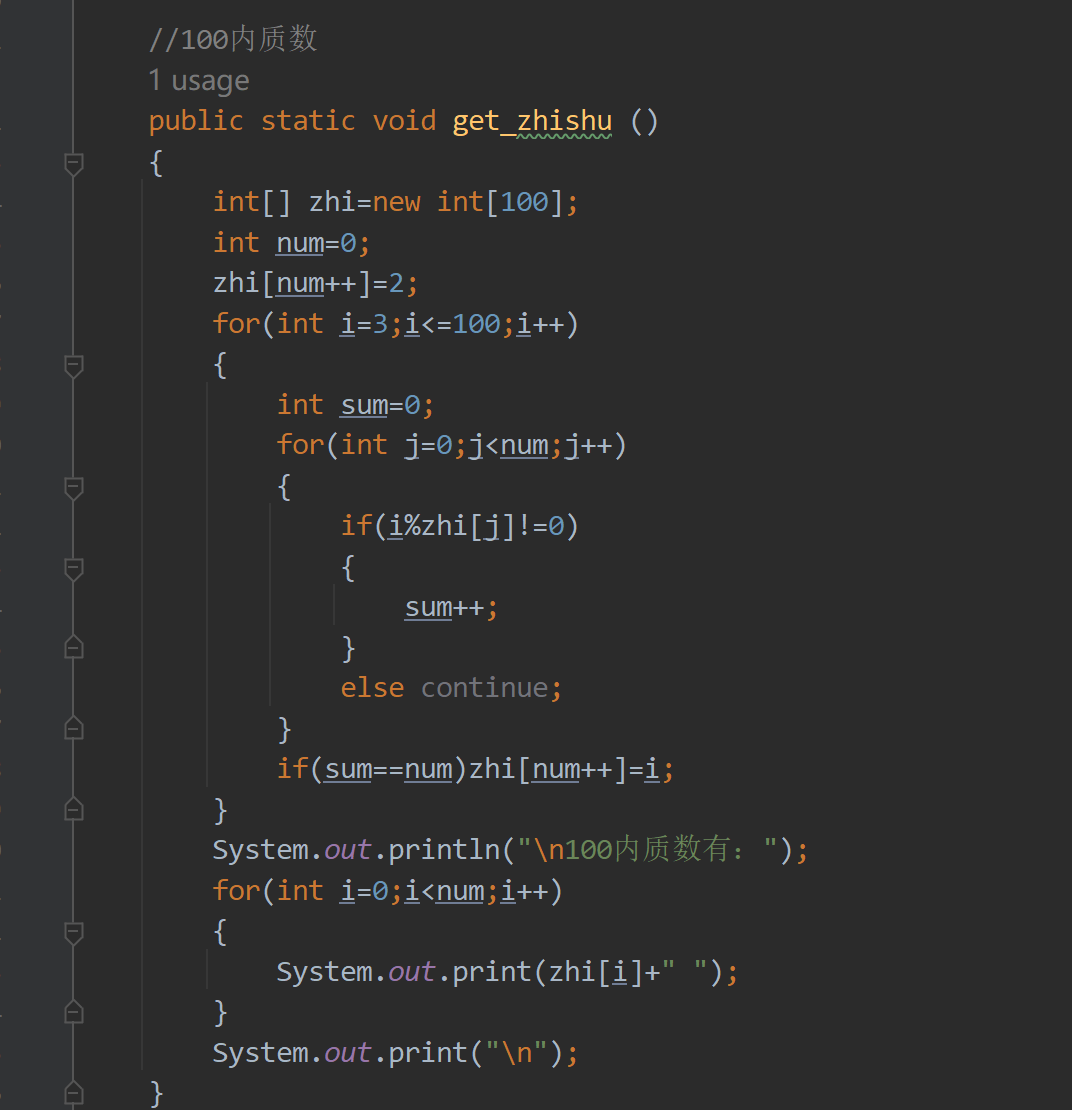




5.



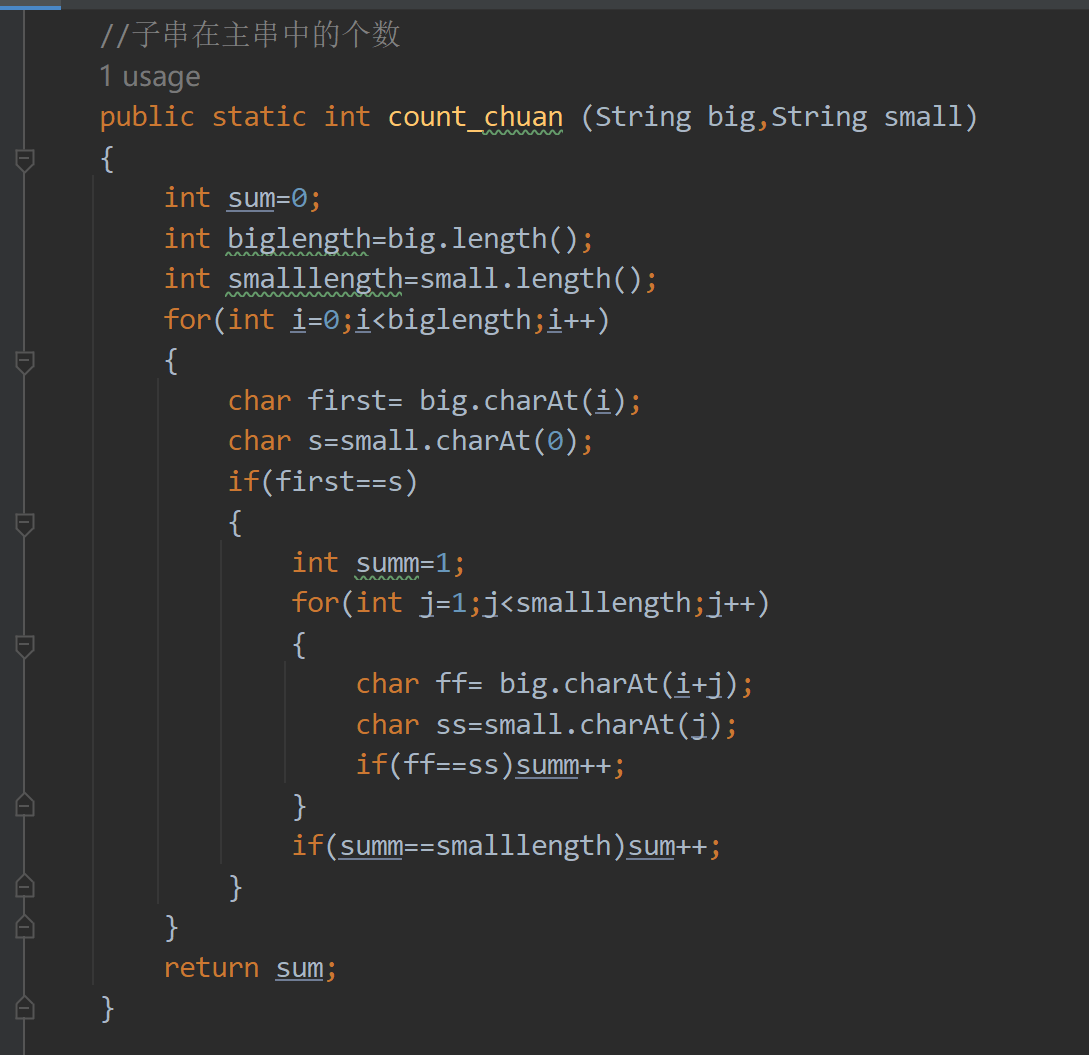
6.



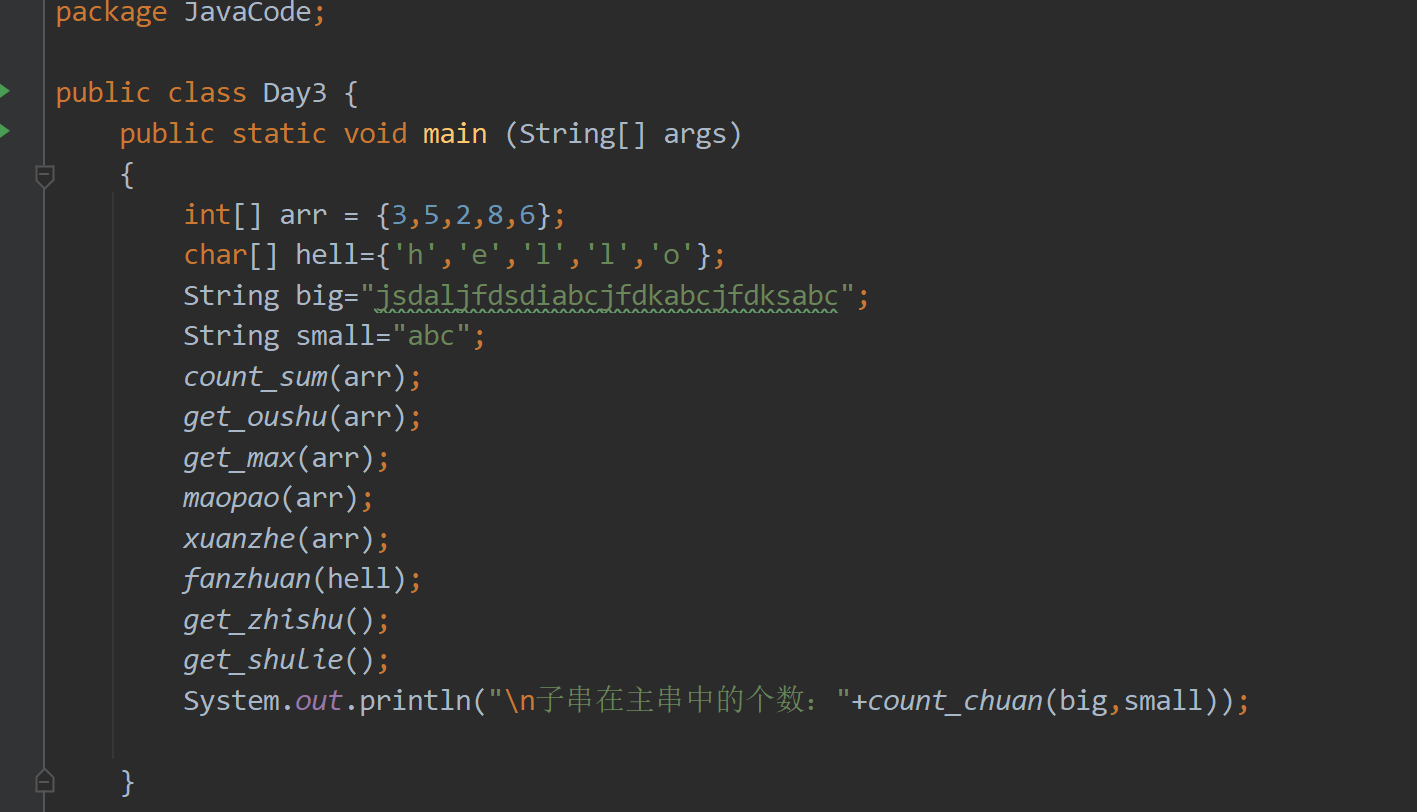
7.



8.



对上述静态方法的调用：



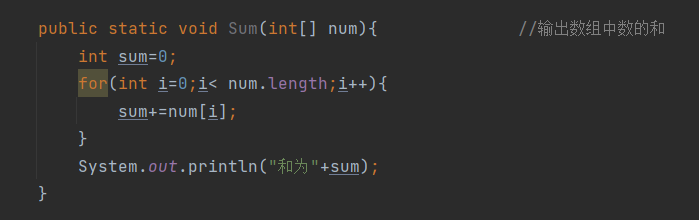
**彭尹垚**

8.17日志

今天学习了Java数组的运用，安装了idea工具，package的使用，学习了在类中包装函数（方法），静态与一般方法的书写结构。分文件编写程序。

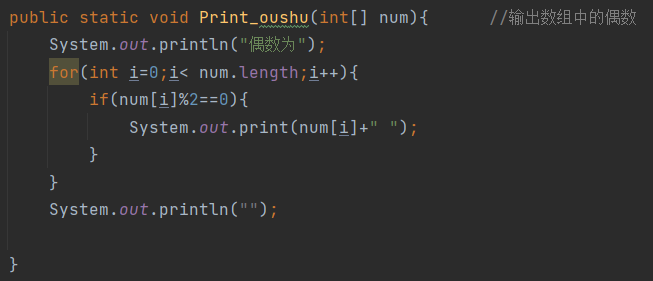
习题：

输出整数数组中所有元素的和



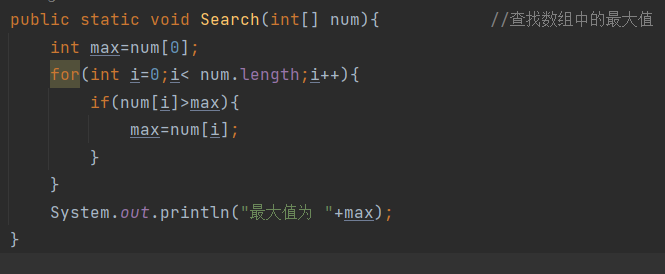


输出数组中为偶数的值



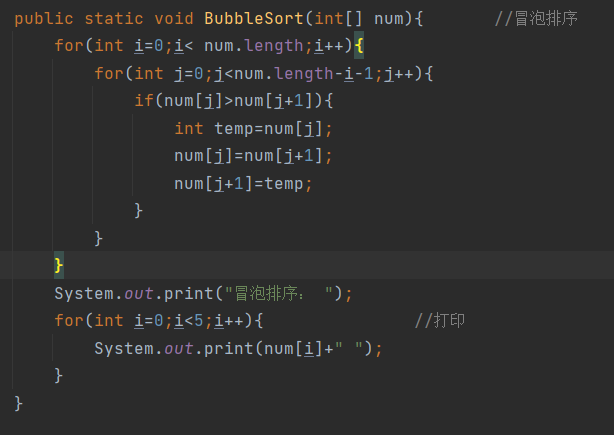


查找到数组中的最大值



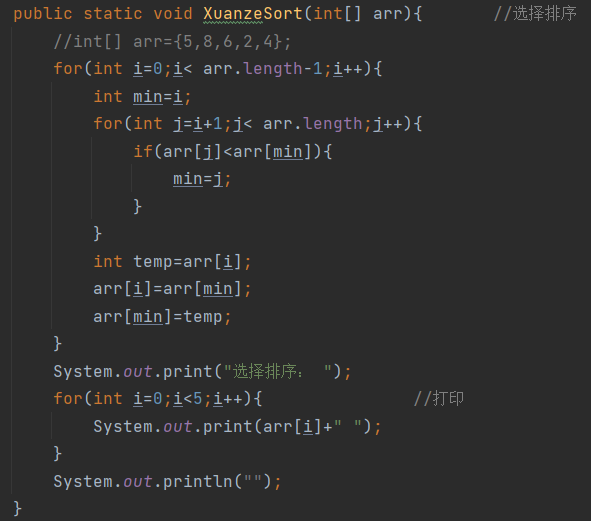


冒泡排序



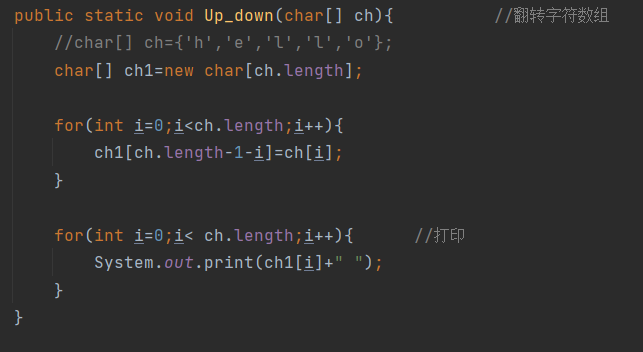


选择排序



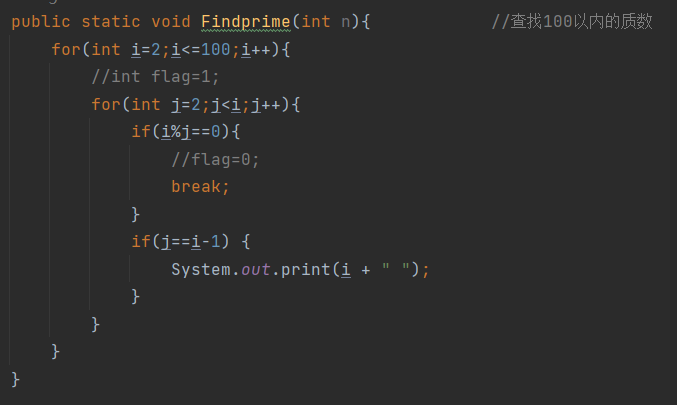


翻转字符数组



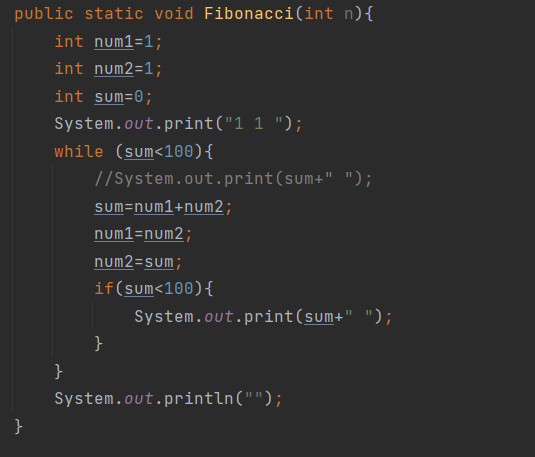


查找n以内的质数



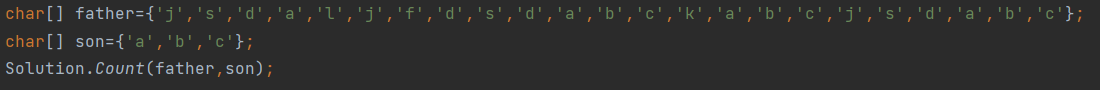


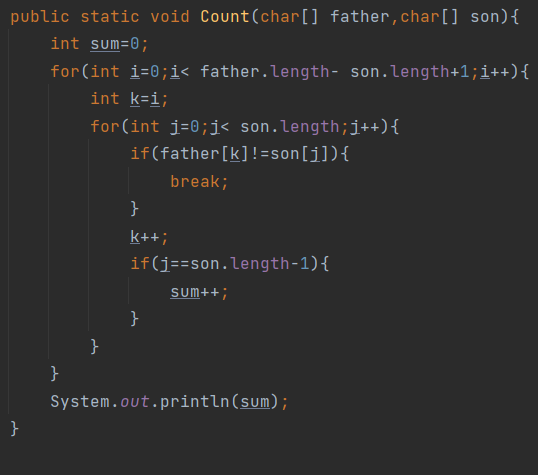
斐波那契数列





序列检测







**谢泽川**

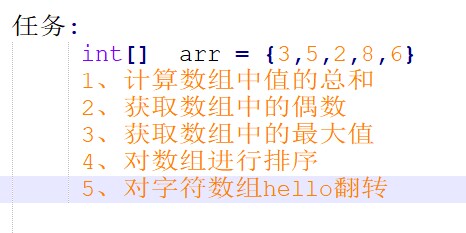
### Java学习日志 - JAVA工具idea的部署和类的编写与数组

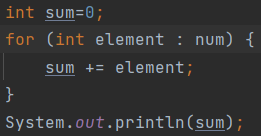
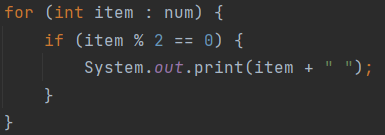
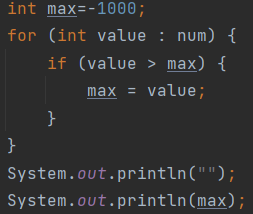
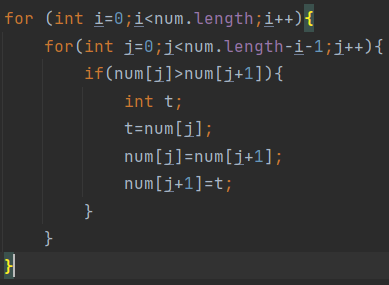
#### 日期：2024年8月17日

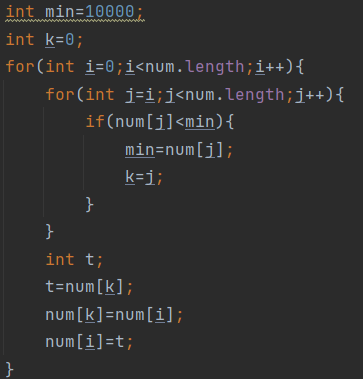
#### 学习内容：

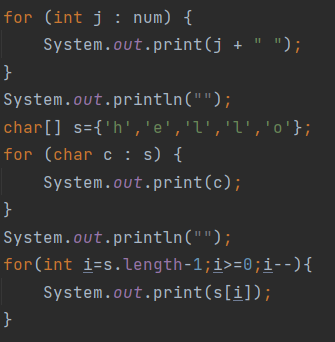
今天的学习主要集中在Java工具的部署和数组的运用和类的编写与打包。

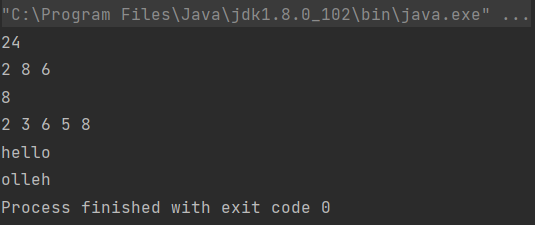
任务一：



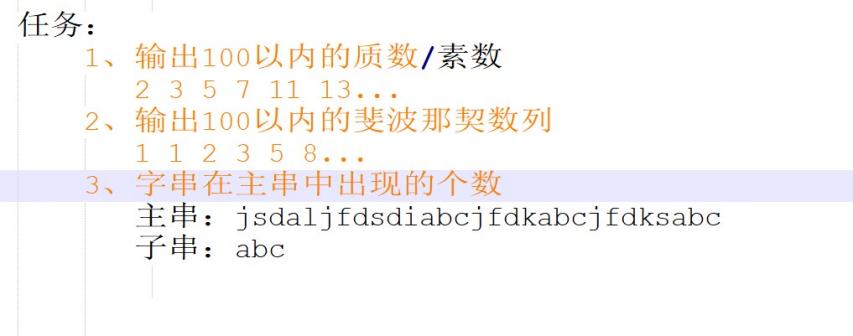
1. 
2. 
3. 
4. 冒泡排序：

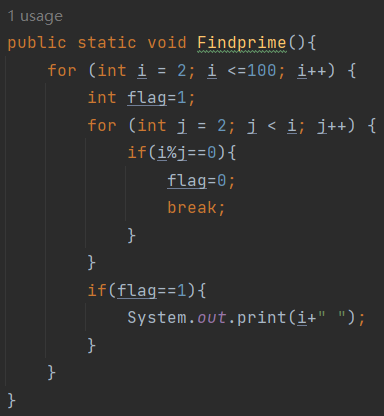
选择排序：

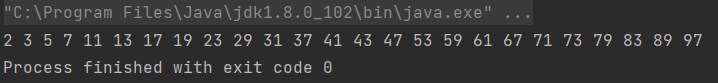
1. 

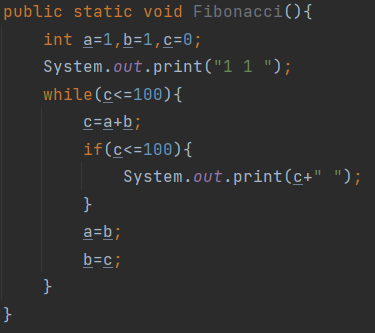


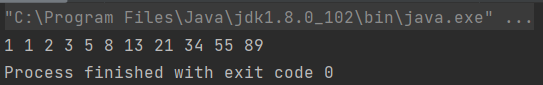
任务二：

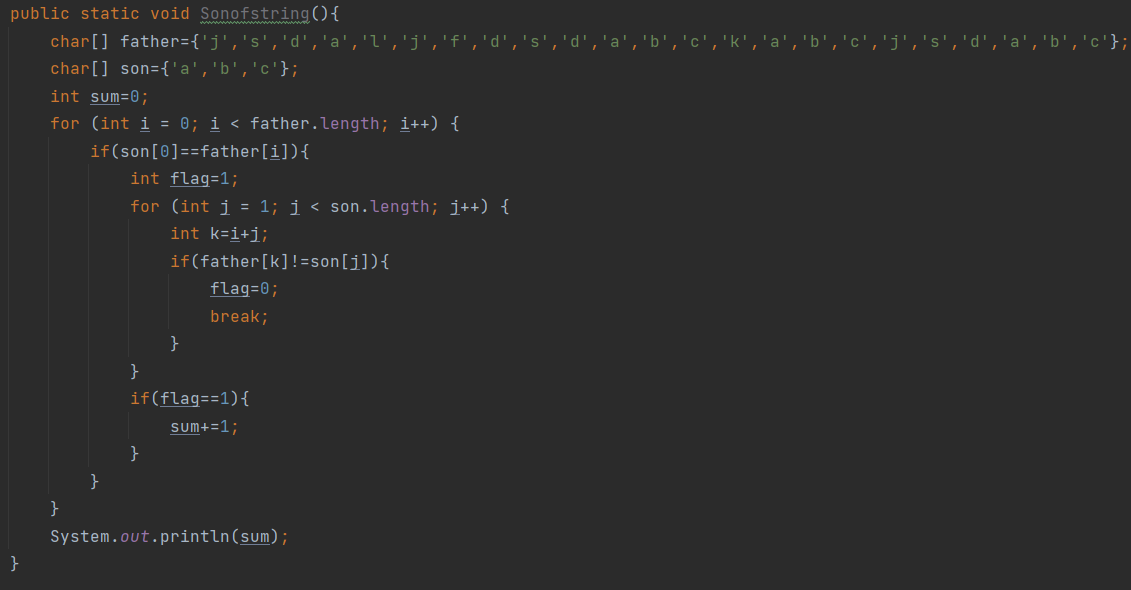


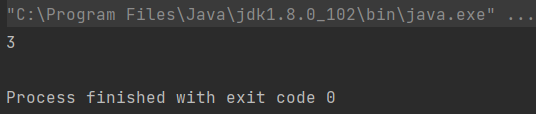
1. 



1. 



1. 

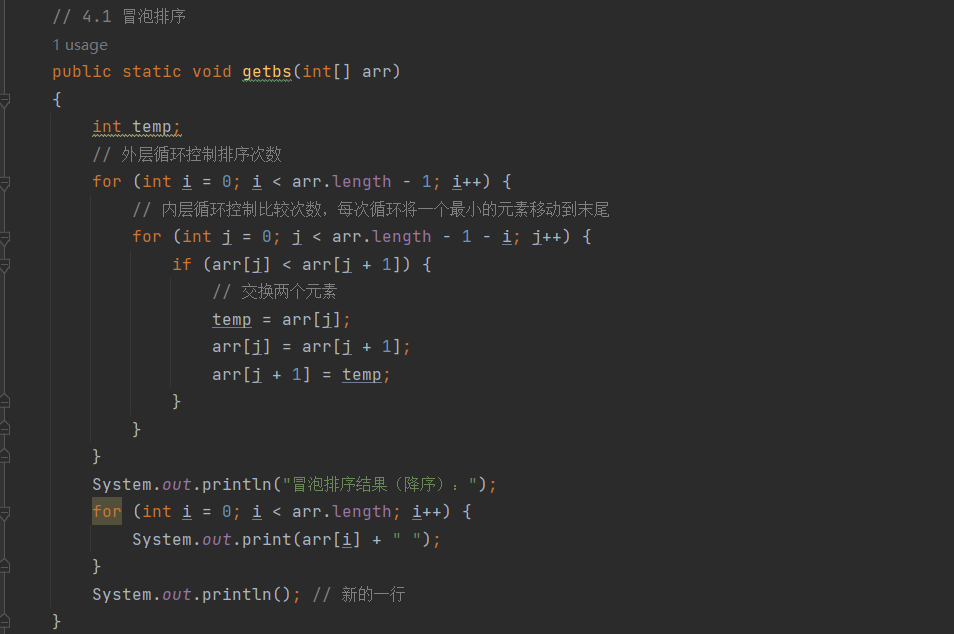


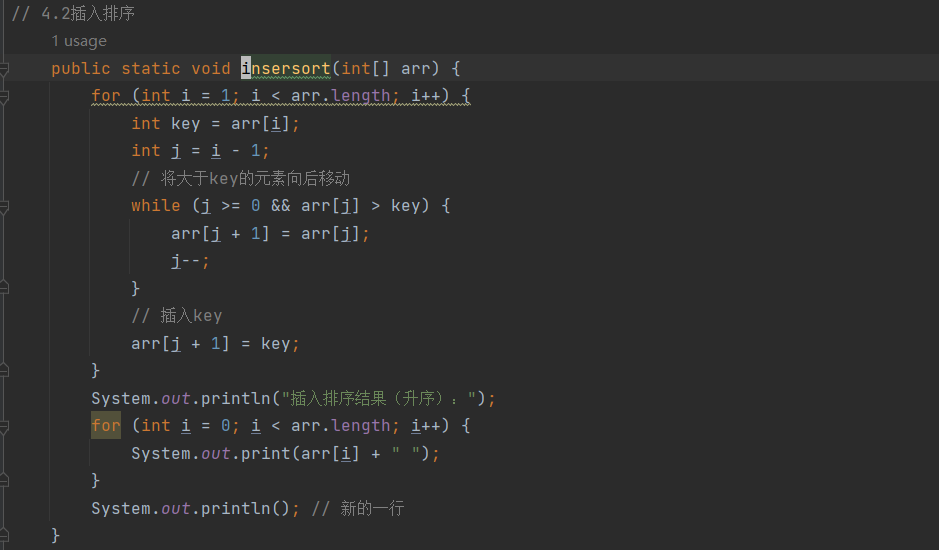
**张世纪**

上午数组五题代码：



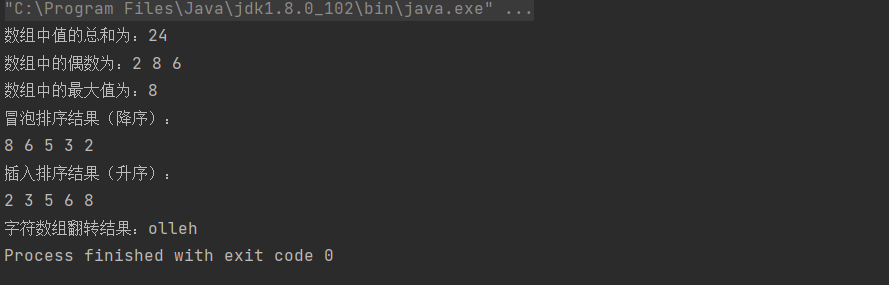




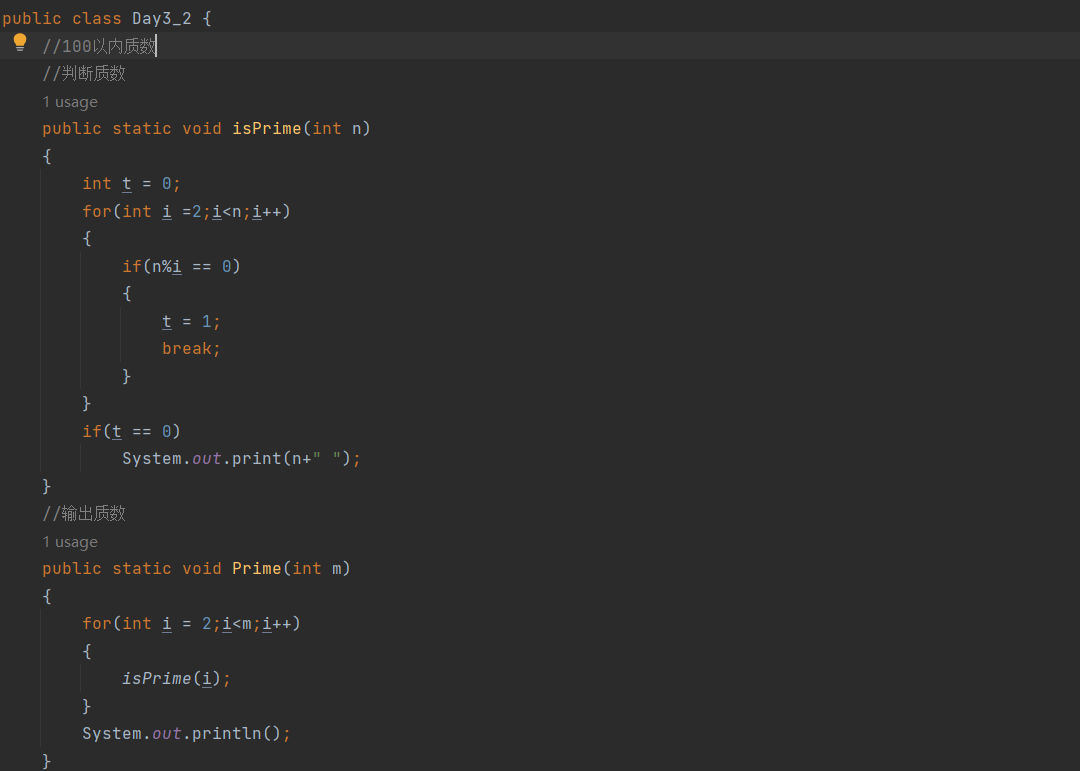


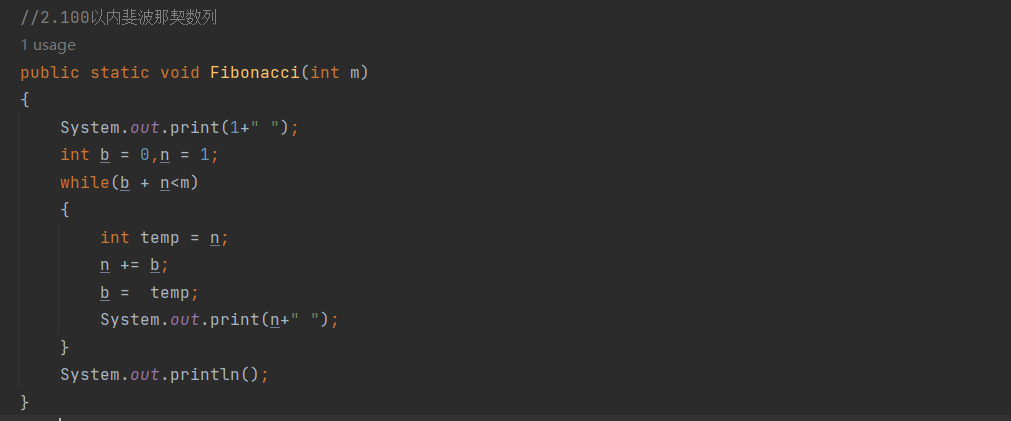


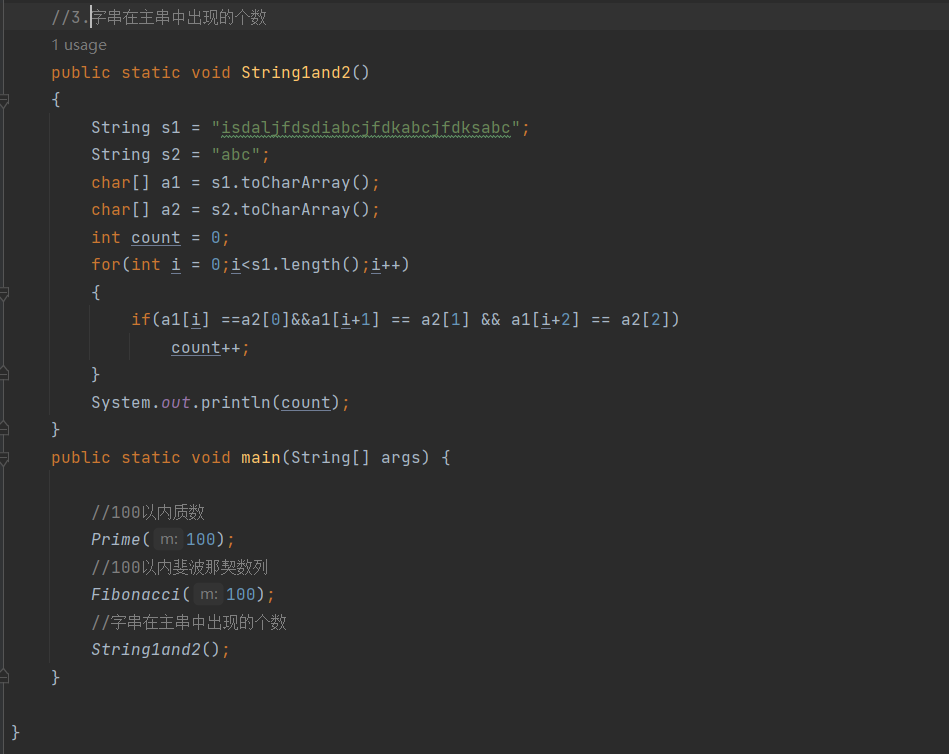
运行结果：



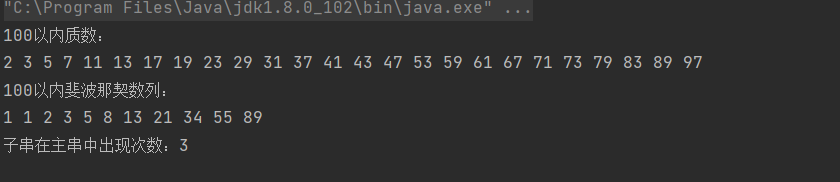
下午三题：







运行结果：



**叶翔昊**

总体情况：继续上课，下载了IDEA软件，熟悉代码编写。布置任务全部完成，问题基本解决。

任务一：int[]arr={3,5,2,8,6}

1.计算数组中值的总和

2.获取数组中的偶数

3.获取数组中的最大值

4.对数组进行排序

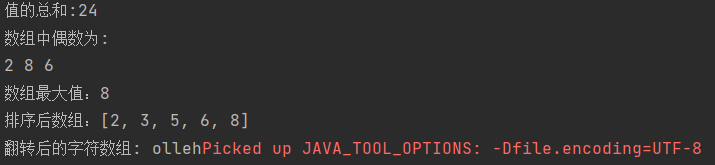
5.对字符数组he11o翻转

完成情况：顺利完成，熟悉数组的用法，复习了排序算法和相关代码编写，

代码：

package com.cq.cd.studnt;  
  
import java.util.Arrays;  
  
public class Main  
{  
 public static void main(String[] args)  
 {  
 //计算数值总和  
 int[] arr = {3, 5, 2, 8, 6};  
 int sum = 0;  
 for (int i = 0; i < 5; i++)  
 {  
 sum += arr[i];  
 }  
 System.*out*.println("值的总和:" + sum);  
 //获取数组中偶数  
 System.*out*.println("数组中偶数为:");  
 for (int i = 0; i < 5; i++)  
 {  
 if (arr[i] % 2 == 0)  
 {  
 System.*out*.print(arr[i] + " ");  
 }  
 }  
 //换行  
 System.*out*.println(" ");  
 //获取数组中最大值  
 int a = 0;  
 for (int i = 0; i < 5; i++)  
 {  
 if (arr[i] > a)  
 {  
 a = arr[i];  
 }  
  
 }  
 System.*out*.println("数组最大值：" + a);  
 for (int i = 0; i < 4; i++)  
 {  
 for (int j = 0; j < 4 - i; j++)  
 {  
 if (arr[j] > arr[j + 1])  
 {  
 int temp = arr[j];  
 arr[j] = arr[j + 1];  
 arr[j + 1] = temp;  
 }  
 }  
 }  
 System.*out*.print("排序后数组：" + Arrays.*toString*(arr));  
  
 System.*out*.println(" ");  
  
 char[] arr1 = {'h', 'e', 'l', 'l', 'o'};  
 // 翻转字符数组  
 int n = arr1.length;  
 for(int i = 0; i<n /2;i++)  
 {  
 char temp = arr1[i];  
 arr1[i] = arr1[n - 1 - i];  
 arr1[n - 1 - i] = temp;  
 }  
  
 // 打印翻转后的字符数组  
 System.*out*.print("翻转后的字符数组: ");  
 for(char c :arr1)  
 {  
 System.*out*.print(c);  
 }  
  
 }  
}

运行情况：



任务二：

输出100以内的质数/素数2 35 7 11 13.

完成情况：顺利完成，熟悉类的建立和使用方法，运用boolean函数

代码：package com.cq.cd.studnt;  
  
public class PrimeNumbers {  
 public static void main(String[] args) {  
 // 创建PrimeNumbers对象  
 PrimeNumbers primeNumbers = new PrimeNumbers();  
  
 // 查找并打印100以内的质数  
 primeNumbers.printPrimes(100);  
 }  
  
 // 打印给定范围内的所有质数  
 public void printPrimes(int max) {  
 System.*out*.println("100以内的质数:");  
 for (int num = 2; num <= max; num++) {  
 boolean isPrime = true; // 假设当前数字是质数  
 if (num <= 1) {  
 isPrime = false; // 1及以下的数字不是质数  
 }  
 for (int i = 2; i <= Math.*sqrt*(num); i++) {  
 if (num % i == 0) {  
 isPrime = false; // 如果能被 i 整除，则 num 不是质数  
 break; // 退出循环  
 }  
 }  
 if (isPrime) {  
 System.*out*.print(num + " ");  
 }  
 }  
 System.*out*.println();  
 }  
}

运行情况：

1723888457496

任务三:

输出100以内的斐波那契数列1 1 2 3 5 8..

完成情况：顺利完成，熟悉类的使用方法和特性

代码：

package com.cq.cd.studnt;  
  
public class FibonacciNumbers {  
 public static void main(String[] args) {  
 // 创建FibonacciNumbers对象  
 FibonacciNumbers fibonacciNumbers = new FibonacciNumbers();  
  
 // 查找并打印100以内的斐波那契数列  
 fibonacciNumbers.printFibonacci(100);  
 }  
  
 // 打印给定范围内的所有斐波那契数  
 public void printFibonacci(int max) {  
 System.*out*.println("100以内的斐波那契数列:");  
  
 int a = 0; // 第一个斐波那契数  
 int b = 1; // 第二个斐波那契数  
  
 // 打印第一个斐波那契数  
 if (a <= max) {  
 System.*out*.print(a + " ");  
 }  
  
 // 打印第二个斐波那契数  
 if (b <= max) {  
 System.*out*.print(b + " ");  
 }  
  
 // 生成后续的斐波那契数  
 while (true) {  
 int next = a + b; // 下一个斐波那契数  
 if (next > max) {  
 break; // 如果下一个斐波那契数超过max，则退出循环  
 }  
 System.*out*.print(next + " ");  
 a = b; // 更新a  
 b = next; // 更新b  
 }  
  
 System.*out*.println();  
 }  
  
}

运行情况：

1723888515680

任务四：

字串在主串中出现的个数主串:jsdaljfdsdiabcjfdkabcjfdksabc字串:abc

完成情况：顺利完成。熟悉相关函数使用方法

代码：

package com.cq.cd.studnt;  
  
public class SubstringCounter {  
 public static void main(String[] args) {  
 // 创建SubstringCounter对象  
 SubstringCounter counter = new SubstringCounter();  
  
 // 定义主串和字串  
 String mainString = "jsdaljfdsdiabcjfdkabcjfdksabc";  
 String subString = "abc";  
  
 // 调用方法计算字串在主串中出现的次数  
 int count = counter.countSubstringOccurrences(mainString, subString);  
  
 // 打印结果  
 System.*out*.println("字串 \"" + subString + "\" 在主串中出现的次数: " + count);  
 }  
  
 // 计算字串在主串中出现的次数  
 public int countSubstringOccurrences(String mainString, String subString) {  
 if (mainString == null || subString == null || subString.length() == 0) {  
 return 0; // 如果主串或字串为空，或字串长度为0，则返回0  
 }  
  
 int count = 0;  
 int index = 0;  
  
 // 循环查找字串在主串中的位置  
 while ((index = mainString.indexOf(subString, index)) != -1) {  
 count++; // 找到一次，计数器加1  
 index += subString.length(); // 更新索引位置到下一个字串的起始位置  
 }  
  
 return count;  
 }  
}

运行情况：

1723888593829

**兰天阳**

2024.8.17

创建项目：

项目层级：project->module->package->class

module按照域名命名，package按照域名倒序命名

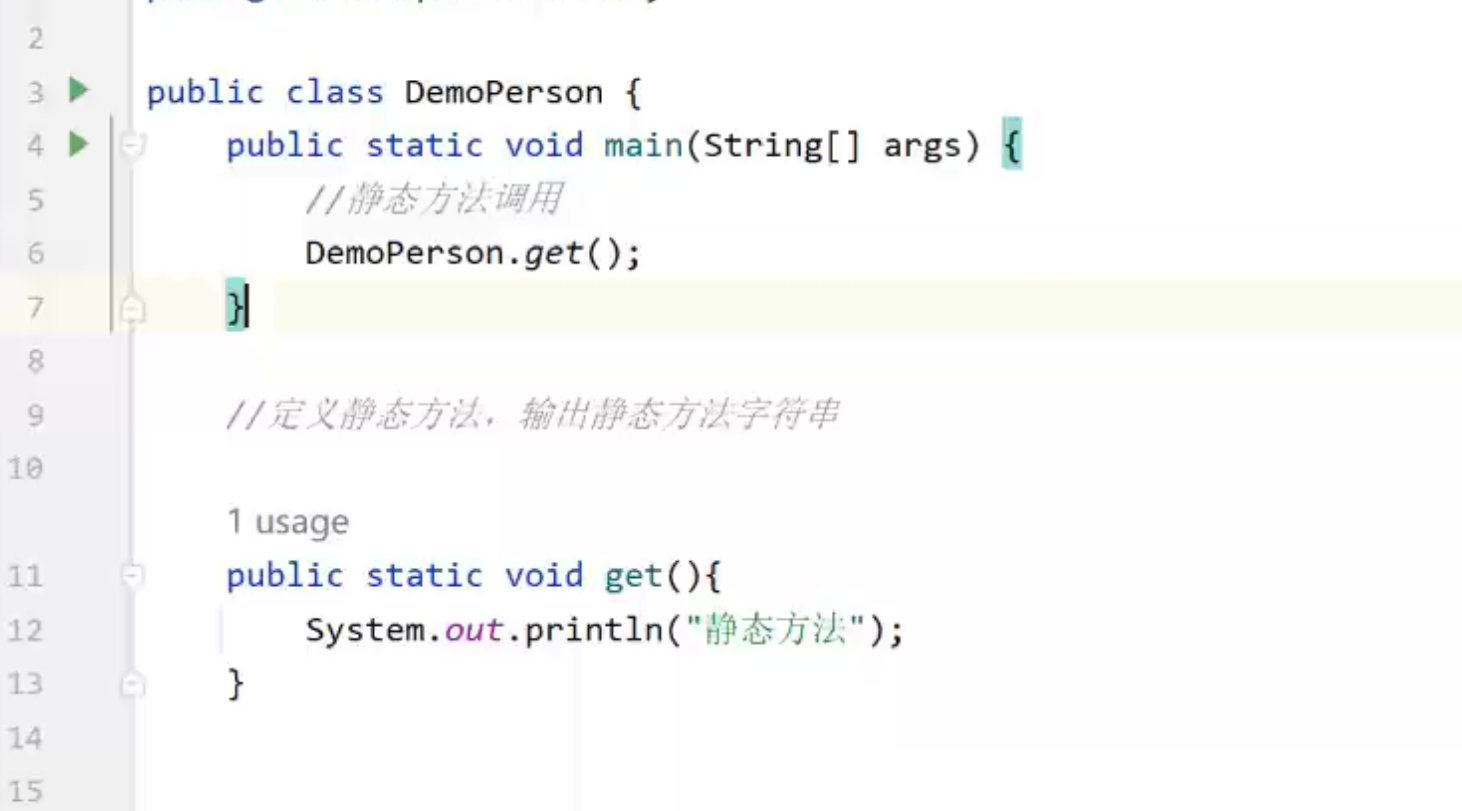
数组：

创建数组：int[] num = new int [5];

同时赋值：int[] num = new int []{1,2,3,4};

成员方法属于对象，调用对象

静态方法属于类，调用类名+静态方法名称，或直接调用静态方法名称



上午任务：

package cd;

public class Day3\_1 {

//求和

public static void sum(int[] arr){

int sum = 0;

for(int i = 0; i < arr.length; i++){

sum += arr[i];

}

System.out.println("总和: "+sum);

}

//找出最大的数

public static void find\_even\_number(int[] arr){

System.out.print("偶数: ");

for(int i = 0; i < arr.length-1; i++){

if(arr[i] %2 == 0){

System.out.print(arr[i] + " ");

}

}

System.out.println(arr[arr.length-1]);

}

public static void find\_max\_number(int[] arr){

int max = 0;

for(int i = 0; i < arr.length; i++){

if(arr[i] >= max){

max = arr[i];

}

}

System.out.println("最大值: "+max);

}

//冒泡排序

public static void bubble\_sort(int[] arr){

int temp = 0;

for(int i = 0; i < arr.length-1; i++){

for(int j = 0; j < 4 - i; j++){

if(arr[j] > arr[j+1]){

temp = arr[j];

arr[j] = arr[j+1];

arr[j+1] = temp;

}

}

}

System.out.print("冒泡排序：");

for(int i = 0; i < arr.length-1; i++){

System.out.print(arr[i] + " ");

}

System.out.println(arr[4]);

}

//快速排序

public static void quicksort(int[] arr, int low, int high){

int i, j, temp, t;

if(low > high){

return;

}

i = low;

j = high;

temp = arr[low];

while(i < j){

while(temp <= arr[j] && i < j){

j--;

}

while(temp >= arr[i] && i < j){

i++;

}

if(i < j){

t = arr[j];

arr[j] = arr[i];

arr[i] = t;

}

}

arr[low] = arr[i];

arr[i] = temp;

quicksort(arr, low, j-1);

quicksort(arr, j+1, high);

}

//翻转字符数组

public static void reverse(char[] h){

for(int i = 0; i < h.length - 1; i++){

System.out.print(h[i]);

}

System.out.println(h[h.length-1]);

char t;

for(int i = 0; i < h.length/2; i++){

t = h[i];

h[i] = h[h.length - i -1];

h[h.length - i -1] = t;

}

for(int i = 0; i < h.length - 1; i++){

System.out.print(h[i]);

}

System.out.println(h[h.length-1]);

}

public static void main(String[] args){

int[] arr = {3, 5, 2, 8, 6};

sum(arr);//求和

find\_even\_number(arr);//寻找偶数

find\_max\_number(arr);//找出最大的数

bubble\_sort(arr);//冒泡排序

quicksort(arr, 0, arr.length-1); //快速排序

System.out.print("快速排序：");

for(int k = 0; k < arr.length-1; k++){

System.out.print(arr[k] + " ");

}

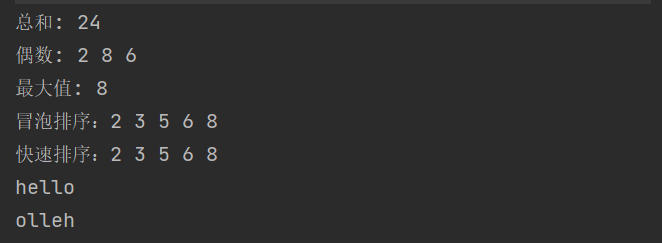
System.out.println(arr[arr.length-1]);

char[] h = {'h', 'e', 'l', 'l', 'o'};

reverse(h);//翻转字符数组hello

}

}



下午任务：

package cd;

public class Day3\_2 {

//寻找素数

public static void find\_prime(int num){

int i;

System.out.print(num+"以内的素数有：");

for(int n = 2; n <= num; n++){

for(i = 2; i <= n; i++){

if(n % i == 0){

break;

}

}

if(i == n){

System.out.print(i + " ");

}

}

}

//寻找斐波那契

public static void find\_fibonacci(int num){

System.out.print(num+"以内的斐波那契数列为：");

for(int i = 1, j = 0, k = 0; i <= num;){

k = i;

System.out.print(k + " ");

i += j;

j = k;

}

}

public static int countstr(String s1, String s2){

int count = 0;

count = countstr(s1, s2, count);

return count;

}

private static int countstr(String s1, String s2, int count){

if(s1.contains(s2)){

count++;

count = countstr(s1.substring(s1.indexOf(s2) + s2.length()), s2, count);

}

return count;

}

public static void main(String[] args){

String s1 = "jsdaljfdsdiabcjfdkabcjfdksabc";

String s2 = "abc";

find\_fibonacci(100);//寻找斐波那契

System.out.println();

System.out.println(countstr(s1, s2));

find\_prime(100);//寻找素数

}

}

