考试：满分 100 分 + 20 分附加题

基础题：5 题（2 分/题，共 10 分）
简单编程题：3 题（5 分/题，15 分）
简单简答题：7 题（5 分/题，共 35 分）
较难编程题：2 题（8 分/题，共 16 分）
较难简答题：3 题（8 分/题，共 24 分）
附加题：2 题（10 分/题，共 20 分）

一、基础题（2 分/题）

1. 用最有效的的方法算出 2 称以 8 等于几

答案：2<<3

2. Math.round(11.5)和 Math.round(-11.5)的值是多少？

Math.round(11.5)：12 Math.round(-11.5)：-11

3. 两个对象 a 和 b，请问 a==b 和 a.equals(b)有什么区别？

a==b：比较对象地址 a.equals(b)：如果 a 对象没有重写过 equals 方法，效果和==相同，如果重写了就按照重写的规则比较。

4. switch 是否能作用在 byte 上，是否能作用在 long 上，是否能作用在 String 上？

答案一：switch 可以作用在 byte 上，不能作用在 long 上，JDK1.7 之后可以作用在 String 上。
答案二：switch 支持的类型 byte,short,int,char，JDK1.5 之后支持枚举，JDK1.7 之后支持 String 类型。

5. char 型变量中是否可以存储一个汉字？

能，因为 Java 一个字符是 2 个字节，每一个字符使用 Unicode 编码表示

6. float f=3.4;是否正确，表达式 15/2*2 的值是多少

答案：不正确，float f = 3.4F;
答案：14

7. 编写代码实现两个变量值交换，int m = 3, n =5;

答案一：
int temp = m;
m = n;
n = temp;

答案二：
m = m + n;
n = m - n;
m = m - n;

答案三：
m = m ^ n;
n = m ^ n;
m = m ^ n;

8. Java 的基本数据类型有哪些？String 是基本数据类型吗？

基本数据类型有：byte,short,int,long,float,double,char,boolean String 是引用数据类型，不是基本数据类型

9. 数组有没有 length()方法？String 有没有 length()方法？File 有没有 length()方法？ArrayList 有没有 length()方法？

数组没有 length()方法，但是有 length 属性。 String 和 File 有 length()方法。 ArrayList 没有 length()方法，有 size()方法获取有效元素个数。

10. String str = new String(“hello”);创建了哪些对象？

字符串常量池中有一个对象，堆中有一个字符串对象。

11. 如何将 String 类型转化 Number 类型？举例说明 String str = “123”;

答任意一个都对：

```
Integer num1 = new Integer(str);
或
int num2 = Integer.parseInt(str);
或
Integer num3 = Integer.valueOf(str);
```

12. 以下代码的运行结果：

```
public static void main(String[] args) {
    char x = 'x';
    int i = 10;
    System.out.println(true? x : i);
    System.out.println(true? 'x' : 10);
}
```

答案：

120

x

```
/*
 * 如果其中有一个是变量，按照自动类型转换规则处理成一致的类型；
 * 如果都是常量，如果一个是 char，如果另一个是[0~65535]之间的整数按 char 处理；
 *          如果一个是 char，另一个是其他，按照自动类型转换规则处理成一致的类型；
 */
```

13. 以下代码的执行结果

```
public static void main(String[] args) {
    int a = 8, b = 3;
    System.out.println(a>>>b);
    System.out.println(a>>>b | 2);
}
```

答案：

1

3

14. 下面程序片段的输出结果是？

```
public static void main(String[] args) {
    int a = 3;
    int b = 1;
    if(a = b){
        System.out.println("Equal");
    }else{
        System.out.println("Not Equal");
    }
}
```

答案：编译不通过

15. 执行如下代码后，c 的值是多少？

```
public static void main(String[] args) {
    int a = 0;
    int c = 0;
    do {
        --c;
        a = a - 1;
    } while (a >= 0);
    System.out.println("c = " + c);
}
```

答案：c = -1

16. 以下代码的运行结果？

```
public static void main(String[] args) {
    int i=10;
    while(i>0){
        i = i +1;
        if(i==10){
            break;
        }
    }
    System.out.println("i=" + i);
}
```

答案一：是一个负数，因为 i 一直累加会超过 int 的存储范围

答案二：死循环

17. 修正如下代码

下面是一段程序，目的是输出 10 个=，但是不小心代码写错了，现在需要修改代码，使得程序完成功能，但是只能“增加”或“修改”其中“一个”字符，很明显，将 i--改为 i++，可以完成功能，但是需要修改“两个”字符，所以并不是一个正确的答案？

```
    public static void main(String[] args) {
        int n=10;
        for (int i = 0; i < n; i--) {
            System.out.println("=");
        }
    }
```

i<n 修改为-i<n

18. 以下代码的运行结果是什么？

```
public class Test {
    public static boolean foo(char c) {
        System.out.print(c);
        return true;
    }

    public static void main(String[] args) {
        int i = 0;
        for (foo('A'); foo('B') && (i < 2); foo('C')) {
            i++;// 1 2
            foo('D');
        }
    }
}
```

答案：ABDCBDCB

19. 以下代码的执行结果是什么

```
    public static void main(String[] args) {
        int i = 0;
        change(i);
        i = i++;
        System.out.println("i = " + i);
    }
    public static void change(int i){
        i++;
```

```
    }
```

答案：i = 0

20. 以下程序的运行结果：

```
    public static void main(String[] args) {
        String str = new String("world");
        char[] ch = new char[]{'h','e','l','l','o'};
        change(str,ch);
        System.out.println(str);
        System.out.println(String.valueOf(ch));
    }
    public static void change(String str, char[] arr){
        str = "change";
        arr[0] = 'a';
        arr[1] = 'b';
        arr[2] = 'c';
        arr[3] = 'd';
        arr[4] = 'e';
    }
```

答案：

world

abcde

21. 以下代码的运行结果是：

```
    public static void main(String[] args) {
        Integer i1 = 128;
        Integer i2 = 128;
        int i3 = 128;
        int i4 = 128;
        System.out.println(i1 == i2);
        System.out.println(i3 == i4);
        System.out.println(i1 == i3);
    }
```

答案：

false

true

true

Integer 的 i1 和 i2 是对象，他们==比较的是地址。如果-128~127 范围，那么使用缓存的常量对象，如果超过这个范围，是新 new 的对象，不是常量对象

22. 以下代码的运行结果

```
    public static void main(String[] args) {
        double a = 2.0;
        double b = 2.0;
        Double c = 2.0;
        Double d = 2.0;
        System.out.println(a == b);
        System.out.println(c == d);
        System.out.println(a == d);
    }
```

答案:

true

false

true

23. 以下代码的运行结果是?

```
public class Test {
    int a;
    int b;
    public void f(){
        a = 0;
        b = 0;
        int[] c = {0};
        g(b,c);
        System.out.println(a + " " + b + " " + c[0]);
    }
    public void g(int b, int[] c){
        a = 1;
        b = 1;
        c[0] = 1;
    }
    public static void main(String[] args) {
        Test t = new Test();
        t.f();
    }
}
```

答案: 1 0 1

24. 以下代码的运行结果是？

```
public class Test {
    static int x, y, z;

    static {
        int x = 5;
        x--;
    }

    static {
        x--;
    }

    public static void main(String[] args) {
        System.out.println("x=" + x);
        z--;
        method();
        System.out.println("result:" + (z + y + ++z));
    }

    public static void method() {
        y = z++ + ++z;
    }
}
```

答案：

x=-1

result:3

25. 以下程序的运行结果是：

```
public class Test {

    public static void main(String[] args) {
        new A(new B());
    }
}
class A{
    public A(){
        System.out.println("A");
    }
    public A(B b){
```

```
        this();
        System.out.println("AB");
    }
}
class B{
    public B(){
        System.out.println("B");
    }
}
```

答案:

B

A

AB

26. 如下代码是否可以编译通过，如果可以，运行结果是什么？

```
interface A{
    int x = 0;
}
class B{
    int x = 1;
}
class C extends B implements A{
    public void printX(){
        System.out.println(x);
    }
    public static void main(String[] args) {
        new C().printX();
    }
}
```

答案：编译错误

System.out.println(x);报错，x 有歧义

27. 以下代码的运行结果：

```
public class Test {


    public static void main(String[] args) {
        Base b1 = new Base();
        Base b2 = new Sub();
    }
```

```
}
class Base{
    Base(){
        method(100);
    }
    public void method(int i){
        System.out.println("base : " + i);
    }
}
class Sub extends Base{
    Sub(){
        super.method(70);
    }
    public void method(int j){
        System.out.println("sub : " + j);
    }
}
```

答案：

```
base : 100
sub : 100
base : 70
```

28. 以下代码的执行过程？

```
    public static void main(String[] args) {
        int test = test(3,5);
        System.out.println(test);
    }

    public static int test(int x, int y){
        int result = x;
        try{
            if(x<0 || y<0){
                return 0;
            }
            result = x + y;
            return result;
        }finally{
            result = x - y;
        }
    }
```

答案：8

29. 以下代码的运行结果？

```
    public static void main(String[] args) {
        Integer[] datas = {1,2,3,4,5};
        List<Integer> list = Arrays.asList(datas);
        list.add(5);
        System.out.println(list.size());
    }
```

运行异常，不允许添加元素

30. 在{1}添加什么代码，可以保证如下代码输出 100

提示：t.wait() 或 t.jion() 或 t.yield() 或 t.interrupt()？

```
public class Test {
    public static void main(String[] args) {
        MyThread m = new MyThread();
        Thread t = new Thread(m);
        t.start();

        ____________{1}_____________

        int j = m.i;
        System.out.println(j);
    }
}
class MyThread implements Runnable{
    int i;
    public void run(){
        try {
            Thread.sleep(1000);
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
        i=100;
    }
}
```

答案：t.join()

31. 以下代码如何优化

```
if(username.equals(“admin”){
   ....
```

}

答案：
if(“admin”.equals(username)){
}

二、基础编程题（5 分/题）

1、用循环控制语句打印输出：1+3+5+...+99=?的结果

答案一：

```
    public static void main(String[] args) {
        int sum = 0;
        for (int i = 1; i <= 99; i+=2) {
            sum += i;
        }
        System.out.println("sum = " + sum);
    }
```

答案二：

```
    public static void main(String[] args) {
        int sum = 0;
        for (int i = 1; i < 100; i++) {
            if (i % 2 != 0) {
                sum += i;
            }
        }
        System.out.println("sum = " + sum);
    }
```

2、请写一个冒泡排序，实现{5,7,3,9,2}从小到大排序

答案一：

```
        int[] arr = { 5, 7, 3, 9, 2 };
        for (int i = 1; i < arr.length; i++) {
            for (int j = 0; j < arr.length - i; j++) {
                if (arr[j] > arr[j + 1]) {
                    int temp = arr[j];
                    arr[j] = arr[j + 1];
                    arr[j + 1] = temp;
                }
            }
        }
```

答案二：

```
        int[] arr = { 5, 7, 3, 9, 2 };
        for (int i = 1; i < arr.length; i++) {
            for (int j = arr.length-1; j >= i; j--) {
                if (arr[j] < arr[j - 1]) {
                    int temp = arr[j];
                    arr[j] = arr[j - 1];
                    arr[j - 1] = temp;
                }
            }
        }
```

3、编写方法实现：求某年某月某日是这一年的第几天

提示：闰年（1）能被 4 整除不能被 100 整除（2）能被 400 整除

```
    public static int daysOfYear(int year, int month, int day){
        int[] daysOfMonth = {31,28,31,30,31,30,31,31,30,31,30,31};
                            补充代码
    }
```

答案一：

```
    public static int daysOfYear(int year, int month, int day){
        int[] daysOfMonth = {31,28,31,30,31,30,31,31,30,31,30,31};
        int sum = day;
        for(int i=0;i< month-1; i++){
            sum += daysOfMonth[i];
        }
        if(month>2){
            if(year%4==0 && year%100!=0 || year%400==0){
                sum++;
            }
        }
        return sum;
    }
```

答案二：

```
    public static int daysOfYear(int year, int month, int day){
        int[] daysOfMonth = {31,28,31,30,31,30,31,31,30,31,30,31};
        int sum = day;
        for(int i=0;i< month-1; i++){
            sum += daysOfMonth[i];
            if(i==1){
                if(year%4==0 && year%100!=0 || year%400==0){
                    sum++;
```

```
                }
            }
        }
        return sum;
    }
```

4、通项公式如下：f(n)=n + (n-1) + (n-2) + ... + 1，其中 n 是大于等于 5 并且小于 10000 的整数，例如：f(5) = 5 + 4 + 3 + 2 + 1，f(10) = 10 + 9 + 8 + 7+ 6 + 5 + 4 + 3 + 2 + 1，请用非递归的方式完成方法 long f(int n)的方法体。

答案一：非递归

```
public static long f(int n) {
    long sum = 0;
    for (int i = 1; i <= n; i++) {
        sum += i;
    }
    return sum;
}
```

5、求 1+2！+3！+...+20！的和

```
public static void main(String[] args)  {
    long sum = 0;
    for (int i = 1; i <= 20; i++) {
        long temp = 1;
        for (int j = 1; j <=i; j++) {
            temp *= j;
        }
        sum += temp;
    }
    System.out.println("sum = " + sum);
}
```

6、输出一个如下图形，一共有 n 行，第 n 行有 2n-1 个*，完成方法 public void printStar(int n)的方法体

```
    *
   ***
  *****
 *******
*********
```

```
    public void printStar(int n) {
        for (int i = 1; i <= n; i++) {
            for (int j = 0; j < n - i; j++) {
                System.out.print(" ");
            }
            for (int j = 0; j < 2 * i - 1; j++) {
                System.out.print("*");
            }
            System.out.println();
        }
    }
```

7、请编写代码使用把一个字符串反转，例如：hello1234，反转后：4321olleh。

答案一：

```
public class Test {
    public static void main(String[] args) {
        String str = "hello1234";
        StringBuilder s = new StringBuilder(str);
        s.reverse();
        str = s.toString();

        System.out.println(str);
    }
}
```

答案二：

```
public class Test {
    public static void main(String[] args) {
        String str = "hello1234";
        char[] array = str.toCharArray();
        for (int i = 0; i < array.length / 2; i++) {
            char temp = array[i];
```

```
            array[i] = array[array.length - 1 - i];
            array[array.length - 1 - i] = temp;
        }
        str = new String(array);
        System.out.println(str);
    }
}
```

8、编写代码实现，从一个标准 url 里取出文件的扩展名，尽可能高效。

```
    public static void main(String[] args) {
        String str = fileExtNameFromUrl("http://localhost:8080/testweb/index.html");
        System.out.println(str);
    }

    public static String fileExtNameFromUrl(String url){
                        补充代码
    }
```

```
    public static void main(String[] args) {
        String str = fileExtNameFromUrl("http://localhost:8080/testweb/index.html");
        System.out.println(str);
    }

    public static String fileExtNameFromUrl(String url){
        return url.substring(url.lastIndexOf('.')+1);
    }
```

9、有一个字符串 String abc = “342567891”，请写程序将字符串 abc 进行升序，可以使用 JDK API 中的现有的功能方法。

参考答案一：

```
public class Test {
    public static void main(String[] args) {
        String str = "342567891";
        char[] arr = str.toCharArray();
        Arrays.sort(arr);
        str = new String(arr);
        System.out.println(str);
```

```
    }
}
```

10、编写一个懒汉式单例设计模式

答案一：懒汉式形式一

```
public class Singleton {
    private static Singleton instance;
    private Singleton(){

    }
    public static Singleton getInstance(){
        if(instance == null){
            synchronized (Singleton.class) {
                if(instance == null){
                    instance = new Singleton();
                }
            }
        }
        return instance;
    }
}
```

答案一：懒汉式形式二

```
public class Singleton{
    private Singleton(){

    }
    private static class Inner{
        private static final Singleton INSTANCE = new Singleton();
    }

    public static Singleton getInstance(){
        return Inner.INSTANCE;
    }
}
```

11、请编写一个饿汉式单例设计模式

答案一：饿汉式形式一

```
public class Singleton {
    public static final Singleton INSTANCE = new Singleton();
    private Singleton(){
```

```
    }
}
```

答案二：饿汉式形式二

```
public class Singleton {
    private static final Singleton INSTACNE = new Singleton();
    private Singleton(){

    }
    public static Singleton getInstance(){
        return INSTACNE;
    }
}
```

答案三：饿汉式形式三

```
public enum Singleton {
    INSTANCE
}
```

12、补充如下枚举类型的代码，使得如下代码达到运行效果

单词提示：monday,tuesday,wednesday,thursday,friday,saturday,sunday

```
import java.util.Scanner;

public class TestWeek {

    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
        System.out.print("今天是星期几(1-7)：");
        int number = input.nextInt();//假设输入的是 2
        Week w = Week.getByNumber(number);
        System.out.println("今天是：" + w);//今天是：TUESDAY(2,星期二)
    }

}
enum Week{
                    (1)

    private int number;
    private String decription;

    private Week(int number, String decription) {
        this.number = number;
        this.decription = decription;
    }
```

```
    public static Week getByNumber(int number){
                    （2）
    }

    @Override
    public String toString() {
        return super.toString()+"(" + number + ","+ decription + ")";
    }
}
```

答案:

```
enum Week{
    MONDAY(1,"星期一"),
    TUESDAY(2,"星期二"),
    WEDNESDAY(3,"星期三"),
    THURSDAY(4,"星期四"),
    FRIDAY(5,"星期五"),
    SATURDAY(6,"星期六"),
    SUNDAY(7,"星期日");

    private int number;
    private String decription;

    private Week(int number, String decription) {
        this.number = number;
        this.decription = decription;
    }

    public static Week getByNumber(int number){
        switch(number){
        case 1:
            return MONDAY;
        case 2:
            return TUESDAY;
        case 3:
            return WEDNESDAY;
        case 4:
            return THURSDAY;
        case 5:
            return FRIDAY;
        case 6:
            return SATURDAY;
        case 7:
            return SUNDAY;
```

```
        default:
            return null;
        }
    }

    @Override
    public String toString() {
        return super.toString()+"(" + number + ","+ decription + ")";
    }
}
```

13、写一段代码实现在遍历 ArrayList 时移除一个元素，例如：”java”？

```
import java.util.ArrayList;
import java.util.Iterator;

public class Test {

    public static void main(String[] args) {
        ArrayList<String> list = new ArrayList<String>();
        list.add("hello");
        list.add("java");
        list.add("world");

                        补充代码
    }
}
```

答案：

```
import java.util.ArrayList;
import java.util.Iterator;

public class Test {

    public static void main(String[] args) {
        ArrayList<String> list = new ArrayList<String>();
        list.add("hello");
        list.add("java");
        list.add("world");

        Iterator<String> iterator = list.iterator();
        while (iterator.hasNext()) {
```

```
            String next = iterator.next();
            if ("java".equals(next)) {
                iterator.remove();
            }
        }
    }
}
```

14、把如下信息添加到 Map 中，并遍历显示，请正确指定泛型

浙江省
    绍兴市
    温州市
    湖州市
    嘉兴市
    台州市
    金华市
    舟山市
    衢州市
    丽水市
海南省
    海口市
    三亚市
北京市
    北京市

参考答案一：

```
import java.util.Arrays;
import java.util.HashMap;
import java.util.List;
import java.util.Map.Entry;
import java.util.Set;

public class Test {

    public static void main(String[] args) throws Exception {
        HashMap<String,List<String>> map = new HashMap<String,List<String>>();
        map.put("北京市", Arrays.asList("北京市"));
        map.put("海南省", Arrays.asList("海口市","三亚市"));
        map.put("浙江省", Arrays.asList("绍兴市","温州市","湖州市","嘉兴市","台州市","
金华市","舟山市","衢州市","丽水市"));
```

```
        Set<Entry<String, List<String>>> entrySet = map.entrySet();
        for (Entry<String, List<String>> entry : entrySet) {
            System.out.println(entry.getKey());
            List<String> value = entry.getValue();
            for (String string : value) {
                System.out.println("\t" + string);
            }
        }
    }
}
```

参考答案二：

```
import java.util.ArrayList;
import java.util.HashMap;
import java.util.Map.Entry;
import java.util.Set;

public class Test {

    public static void main(String[] args) throws Exception {
        HashMap<String,ArrayList<String>> map = new
HashMap<String,ArrayList<String>>();

        ArrayList<String> bj = new ArrayList<String>();
        bj.add("北京市");
        map.put("北京市", bj);

        ArrayList<String> hn = new ArrayList<String>();
        hn.add("海口市");
        hn.add("三亚市");
        map.put("海南省", hn);

        ArrayList<String> zj = new ArrayList<String>();
        zj.add("绍兴市");
        zj.add("温州市");
        zj.add("湖州市");
        zj.add("嘉兴市");
        zj.add("台州市");
        zj.add("金华市");
        zj.add("舟山市");
        zj.add("衢州市");
        zj.add("丽水市");
        map.put("浙江省", zj);

        Set<Entry<String, ArrayList<String>>> entrySet = map.entrySet();
```

```
        for (Entry<String, ArrayList<String>> entry : entrySet) {
            System.out.println(entry.getKey());
            ArrayList<String> value = entry.getValue();
            for (String string : value) {
                System.out.println("\t" + string);
            }
        }
    }
}
```

参考答案三：

```
import java.util.HashSet;
import java.util.HashMap;
import java.util.Map.Entry;
import java.util.Set;

public class Test {

    public static void main(String[] args) throws Exception {
        HashMap<String,HashSet<String>> map = new HashMap<String,HashSet<String>>();

        HashSet<String> bj = new HashSet<String>();
        bj.add("北京市");
        map.put("北京市", bj);

        HashSet<String> hn = new HashSet<String>();
        hn.add("海口市");
        hn.add("三亚市");
        map.put("海南省", hn);

        HashSet<String> zj = new HashSet<String>();
        zj.add("绍兴市");
        zj.add("温州市");
        zj.add("湖州市");
        zj.add("嘉兴市");
        zj.add("台州市");
        zj.add("金华市");
        zj.add("舟山市");
        zj.add("衢州市");
        zj.add("丽水市");
        map.put("浙江省", zj);

        Set<Entry<String, HashSet<String>>> entrySet = map.entrySet();
        for (Entry<String, HashSet<String>> entry : entrySet) {
            System.out.println(entry.getKey());
```

```
            HashSet<String> value = entry.getValue();
            for (String string : value) {
                System.out.println("\t" + string);
            }
        }
    }
}
```

15、完成在如下 Map 中查询城市信息

已知有省份 Provice 类型，有属性省份编号 id 和名称 name，有城市 City 类型，有属性城市编号 id 和名称 name,所属省份编号 pid，以及所有信息现保存在一个 Map 中，现在要在 map 中，根据省份编号，查找这个省份下所有的城市。

```
import java.util.HashSet;
import java.util.HashMap;
import java.util.Set;

public class AreaManager {
    private HashMap<Province,HashSet<City>> map;

    public AreaManager(){
        map = new HashMap<Province,HashSet<City>>();

        HashSet<City> bj = new HashSet<City>();
        bj.add(new City(1,"北京市",1));
        map.put(new Province(1,"北京市"), bj);

        HashSet<City> hn = new HashSet<City>();
        hn.add(new City(1,"海口市",2));
        hn.add(new City(2,"三亚市",2));
        map.put(new Province(2,"海南省"), hn);

        HashSet<City> zj = new HashSet<City>();
        zj.add(new City(1,"绍兴市",3));
        zj.add(new City(2,"温州市",3));
        zj.add(new City(3,"湖州市",3));
        zj.add(new City(4,"嘉兴市",3));
        zj.add(new City(5,"台州市",3));
        zj.add(new City(6,"金华市",3));
        zj.add(new City(7,"舟山市",3));
        zj.add(new City(8,"衢州市",3));
        zj.add(new City(9,"丽水市",3));
        map.put(new Province(3,"浙江省"), zj);
```

```
1:北京市
    1:北京市
2:海南省
    1:海口市
    2:三亚市
3:浙江省
    1:绍兴市
    2:温州市
    7:舟山市
    8:衢州市
    9:丽水市
    5:台州市
    4:嘉兴市
    6:金华市
    3:湖州市
```

```
    }

    public HashSet<City> findCity(int pid){
        ____________补充代码____________
    }
}
```

参考答案：

```
    public HashSet<City> findCity(int pid){
        Set<Province> keySet = map.keySet();
        for (Province province : keySet) {
            if(province.getId() == pid){
                return map.get(province);
            }
        }
        return null;
    }
```

16、请编写代码读取一个项目根目录下 info.properties 文件

里面的内容有 user=atguigu 等，请获取 user 的 value 中，并在控制台打印

```
import java.io.FileInputStream;
import java.io.IOException;
import java.util.Properties;

public class Test{
    public static void main(String[] args) throws IOException {
        Properties pro = new Properties();

        //相对于 bin
//      pro.load(ClassLoader.getSystemResourceAsStream("info.properties"));

        //相对于项目根目录
        pro.load(new FileInputStream("info.properties"));

        String username = pro.getProperty("user");
        System.out.println(username);
    }
}
```

17、请编写代码把一个 GBK 的文本文件内容读取后存储到一个 UTF-8 的文本文件中。

```
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.InputStreamReader;
import java.io.OutputStreamWriter;

public class Test {

    public static void main(String[] args) throws Exception {
        FileInputStream fis = new FileInputStream("test_gbk.txt");
        InputStreamReader isr = new InputStreamReader(fis, "GBK");

        FileOutputStream fos = new FileOutputStream("test_utf8.txt");
        OutputStreamWriter oos = new OutputStreamWriter(fos, "UTF-8");

        char[] data = new char[10];
        int len;
        while ((len = isr.read(data)) != -1) {
            oos.write(data, 0, len);
        }

        isr.close();
        fis.close();
        oos.close();
        fos.close();
    }
}
```

18、用实现 Runnable 接口的方式，启动一个线程完成在线程中打印 1-100 的数字

答案一：

```
public class Test {

    public static void main(String[] args) {
        PrintNumberRunnable p = new PrintNumberRunnable();
        Thread t = new Thread(p);
        t.start();
    }
```

```
}
class PrintNumberRunnable implements Runnable{

    @Override
    public void run() {
        for(int i=1; i<=100; i++){
            System.out.println("i=" + i);
        }
    }

}
```

答案二：

```
public class Test {

    public static void main(String[] args) {
        new Thread(new Runnable(){
                @Override
                public void run() {
                    for(int i=1; i<=100; i++){
                        System.out.println("i=" + i);
                    }
                }
        }).start();
    }
}
```

三、基础简答题（5 分/题）

1、break、continue、return 的区别？

break 用于 switch 和循环，用于结束 switch，和当前循环
continue 用于循环，用于结束本次循环
return 用于结束当前方法，还可以用于 return 返回值;返回结果

2、请列出一些常用的类、接口、包，各至少 5 个

注意答案不固定
常用类：String，Math，,ArrayList，HashMap，System
常用接口：Comparable，Comparator，Runnable，Serializable，Collection

常用包：java.lang， java.util，java.io，java.net，java.text，java.lang.reflect

3、访问修饰符的作用范围由大到小，及各自的范围是什么？可以修饰什么？

public->protected->缺省(default)->private

修饰符	类内部	同一个包	子类	任何地方
private	Yes			
(缺省)	Yes	Yes		
protected	Yes	Yes	Yes	
public	Yes	Yes	Yes	Yes

外部类只能使用 public 或缺省。
如果是修饰类的成员，四种都可以。

4、请对 public static void main(String[] args)的每一个单词做解释？

public：公共的，用它修改的类或成员在任意位置可见
static：静态的，用它修改的方法，可以不用创建对象就可以调用
void：表示该方法没有返回值
main：Java 的主方法名，JavaSE 的程序入口
String[]：字符串数组，这是 main 方法的形参类型，可以通过命令行参数传值
args：这是 main 方法的形参名，如果要在 main 中使用命令行参数，可以遍历该 args 数组。

5、请解释 Overload 与 Override 的区别？

Overload 是方法重载，指的是在同一个类中，方法名称相同，形参列表不同的两个或者多个方法，和返回值类型无关。
Override 是方法的重写，指的是子类在继承父类时，当父类的方法体不适用于子类时，子类可重写父类的方法。重写必须遵守方法名和形参列表与父类的被重写的方法相同，而返回值类型可以小于等于父类被重写的方法（如果是基本数据类型和 void 必须相同），权限修饰符可以大于等于父类被重写的方法，抛出的异常列表可以小于等于父类被重写的方法。

6、final、finalize、finally 的区别？

final 是表示最终的，是一个修饰符，修饰类时表示不能被继承，修饰方法时表示不能被子类

重写，修饰属性和局部变量时表示值不能被修改，是个常量。
finally 是表示最终块，是异常处理的一部分，和 try..catch 一起使用，不管是否发生异常都要执行的代码放在 finally 块中。
finalize 是表示最终方法，是 java.lang.Object 类的一个方法，在对象被垃圾回收时调用。

7、面向对象的基本特征有哪些？并作出解释

答出三个基本特征给 3 分
面向对象的基本特征有：
（1）封装：封装的好处就是安全，方便。封装隐藏了对象的具体实现，当要操纵对象时，只需调用其中的方法，而不用管方法的具体实现。属性的封装就是属性私有化并提供 get/set 方法，这样外界只能通过 get/set 方法来操作属性，行为变得可控。
（2）继承：继承的好处就是代码的复用和扩展。继承可以保留父类的属性和方法，同时子类又可以扩展自己的属性和方法。
（3）多态：目的是实现代码的灵活性，多态体现在重载和重写方法，更多的时候指的是对象的多态性，即当父类的变量指向子类的对象时，那么调用子类重写的方法时，运行的是子类重写过的代码，从而实现同一个父类的变量，因为赋值的子类对象不同而体现出不同的功能。应用主要体现在多态参数和多态数组中。

8、请解释 String、StringBuilder、StringBuffer 的区别？

String 是不可变的字符序列，因此字符串常量存储在常量池中，而 StringBuilder 和 StringBuffer 是可变的字符序列。
String 对象是常量对象，因此一旦拼接和修改就会产生新的 String 对象。
SringBuffer 和 StringBuilder 可以在原对象上进行 append,insert,delete,replace 等修改。
StringBuilder 和 StringBuffer 是完全兼容的 API，但是 StringBuilder 是线程不安全的、StringBuffer 是线程安全的。

9、如下关于 String 比较的代码的运行结果是什么

```
    public static void main(String[] args) {
        String str1 = "1";
        String str2 = "2";
        String str3 = new String("1");
        final String str4 = "2";
        final String str5 = new String("2");
        String str6 = "12";

        String str7 = "1" + "2";
        String str8 = str1 + "2";
        String str9 = str1 + str2;
        String str10 = str3 + str4;
```

```
        String str11 = "1" + str4;
        String str12 = "1" + str5;
        String str13 = (str1 + str2).intern();

        System.out.println("(1)"+ (str1 == str3));
        System.out.println("(2)"+ (str2 == str4));
        System.out.println("(3)"+ (str4 == str5));
        System.out.println("(4)"+ (str6 == str7));
        System.out.println("(5)"+ (str6 == str8));
        System.out.println("(6)"+ (str6 == str9));
        System.out.println("(7)"+ (str6 == str10));
        System.out.println("(8)"+ (str6 == str11));
        System.out.println("(9)"+ (str6 == str12));
        System.out.println("(10)"+ (str6 == str13));
    }
```

答案：一个 0.5 分

```
(1)false
(2)true
(3)false
(4)true
(5)false
(6)false
(7)false
(8)true
(9)false
(10)true
```

10、BigDecimal 和 float、double 有什么区别？BigInteger 和 int、long 有什么区别？

在用 C 或者 C++处理大数时感觉非常麻烦，但是在 Java 中有两个类 BigInteger 和 BigDecimal 分别表示大整数类和大浮点数类，至于两个类的对象能表示最大范围不清楚，理论上能够表示无线大的数，只要计算机内存足够大。这两个类都在 java.math.*包中，因此每次必须在开头处引用该包。

BigInteger 和 BigDecimal 是用对象表示数据的，其实底层是用字符串存储数据的，因此无法使用“算术运算符”进行算术运算，只能调用 add 等方法完成计算。

而 float,double,int,long 等是基本数据类型，可以直接用算术运算符运算，但是有存储范围有限以及精度的问题。

11、请对 Java 的基本数据类型与包装类做解释？

Java 的八种基本数据类型与包装类：

byte <--> Byte
short <-->Short
int <--> Integer
long <--> Long
float <--> Float
double <--> Double
char <--> Character
boolean <--> Boolean
八种基本数据类型只与自己的包装类之间进行装箱与拆箱。JDK1.5 之后支持自动装箱与自动拆箱。

12、java.lang.Comparable 与 java.util.Comparator 有什么区别？

java.lang.Comparable<T>被称为自然排序接口。包含一个抽象方法 int compareTo(T obj)，如果当前对象比指定对象 obj 大，则返回正整数，小则返回负整数，相等返回 0。
java.util.Comparator<T>被称为定制排序接口。包含一个抽象方法 int compare(T t1, T t2)，如果 t1 大于 t2，则返回正整数，t1 小于 t2，则返回负整数，相等返回 0。
如果在使用 Arrays.sort(数组)或 Collections.sort(Collection 集合)方法时，TreeSet 和 TreeMap 时元素默认按照 Comparable 比较规则排序；也可以单独为 Arrays.sort(数组)或 Collections.sort(Collection 集合)方法时，TreeSet 和 TreeMap 指定 Comparator 定制比较器对象。

13、请解释 Collection 和 Collections 的区别?List、Set、Map 是否继承 Collection？

Collection 是接口，是 List 和 Set 系列接口的父接口。是 Collection 系列接口的根接口。
Collections 是工具类，其中提供了很多静态方法来操作各种集合。
List 和 Set 继承 Collection，Map 不继承 Collection。

14、请解释 Arraylist、Linkedlist 和 Vector 的区别？

ArrayList：是线程不安全的动态数组，底层是数组结构，JDK1.7 后初始化为空数组，在添加第一个元素时初始化为长度为 10 的数组，如果容量满了，按照 1.5 倍扩容。支持 foreach 和 Iterator 遍历。
Vector：是线程安全的动态数组，底层是数组结构，初始化为长度为 10 的数组，如果容量满了，按照 2.0 倍扩容。除了支持 foreach 和 Iterator 遍历，还支持 Enumeration 迭代。
LinkedList：是双向链表，底层是链表结构。当频繁在集合中插入、删除元素时，效率较高，但是查找遍历的效率较低。

15、Hashtable 与 HashMap 的区别？如何解决那个线程不安全的问题？

Hashtable 是线程安全的哈希表，底层结构是数组+链表。
HashMap 是线程不安全的哈希表，底层结构是 JDK1.7 时数组+链表，JDK1.8 时数组+链表/红黑树。
HashMap 的线程安全问题可以使用 Collections 的 synchronizedMap(Map<K,V> m) 方法解决。

16、List、Map、Set 三个接口，存取元素时，各有什么特点？

List：是有序的，可重复的，添加元素的方法是 add，可以根据索引获取元素。
Set：是无序的，不可重复的，添加元素的方式是 add，HashSet 和 LinkedHashSet 的元素是依据 hashCode 和 equals 区别元素是否相等，而 TreeSet 是依据 compareTo 或 compare 区别元素是否相等。
Map：是存储键值对的，添加的方法是 put(key,value)，可以根据 key 获取 value。

17、ArrayList 和 LinkedList 的底层实现（存储结构、扩容机制）

1.ArrayList 是实现了基于动态数组的数据结构，LinkedList 基于链表的数据结构。
2.对于随机访问 get 和 set，ArrayList 觉得优于 LinkedList，因为 LinkedList 要移动指针。
3.对于新增和删除操作 add 和 remove，LinedList 比较占优势，因为 ArrayList 要移动数据。 这一点要看实际情况的。若只对单条数据插入或删除，ArrayList 的速度反而优于 LinkedList。但若是批量随机的插入删除数据，LinkedList 的速度大大优于 ArrayList. 因为 ArrayList 每插入一条数据，要移动插入点及之后的所有数据。

18、请列举一些常见的异常或错误类型（至少 5 个）

运行时异常：
　　数组下标越界异常：ArrayIndexOutOfBoundsException
　　类型转换异常：ClassCastException
　　算术异常：ArithmeticException
　　空指针异常：NullPointerException
编译时异常：
　　IO 操作异常：IOException
　　文件找不到异常：FileNotFoundException
　　已到达文件流末尾异常：EOFException

类找不到异常：ClassNotFoundException
没有对应的方法异常：NoSuchMethodException
错误：
堆内存溢出：OutOfMemoryError
栈内存溢出：StackOverflowError

19、请解释 Java 异常处理的过程

Java 的异常处理过程如下：
（1）当程序运行到某一句代码，如果发生了异常（可能是 JVM 判定的异常，也可能是遇到 throw 的），程序都会停下来，然后把异常信息封装到异常的对象中，并且“抛”出
（2）JVM 会检测在这段程序代码的外围，是否有 try...catch，如果有 try...catch，就判断是否有 catch 可以捕获它，如果捕获了，程序就进入对应的 catch 块进行异常处理，处理后程序继续运行 try..cath 之后的代码。
（3）JVM 会检测在这段程序代码的外围，根本就没有 try...catch 或者是有 try...catch 但是捕获不住，即类型对不上，JVM 都会把这个异常对象抛出“上级，方法的调用者”
（4）上级一旦接到异常对象，处理过程还是 1,2,3
（5）如果一直抛，一路上都没有可以捕获它，程序就崩溃了。

20、请解释 Java 异常处理机制相关的 5 个关键字

try：尝试执行可能发生异常的代码。
catch：尝试捕获 try 部分发生的异常。可以存在多个 catch，如果多个 catch 的异常类型有继承关系，那么遵循子上父下。
finally：不管是否发生异常都要执行的代码放在 finally 块中。
throws：方法声明时显示抛出异常，指定该方法可能抛出的异常类型列表。
throw：手动抛出异常，可以抛出系统预定异常，也可以抛出用户自定异常，而且用户自定义异常必须用 throw 语句抛出，可以代替 return 语句结束方法运行。

21、Java 中的 IO 流的四大基类是什么（2 分）,请列出常用的 IO 流类型（至少 5 个）（3 分）

所有的 IO 流都是从以下四个抽象基类，超级父类中分出来的：
（1）字节输入流：InputStream
（2）字节输出流：OutputStream
（3）字符输入流：Reader
（4）字符输出流：Writer

可以延伸出很多 IO 流，例如：和文件相关
（1）文件字节输入流：FileInputStream
（2）文件字节输出流：FileOutputStream

（3）文件字符输入流：FileReader
（4）文件字符输出流：FileWriter

例如：缓冲流
（1）字节输入缓冲流：BufferedInputStream
（2）字节输出缓冲流：BufferedOutputStream
（3）字符输入缓冲流：BufferedReader
（4）字符输出缓冲流：BufferedWriter

例如：转换流
（1）InputStreamReader：把字节输入流转为字符输入流，解码
（2）OutputStreamWriter：把字符输出流转为字节输出流，编码

例如：数据流
（1）字节输入数据流：DataInputStream
（2）字节输出数据流：DataOutputStream

例如：对象流
（1）对象输入流：ObjectInputStream，用于对象的序列化
（2）对象输出流：ObjectOutputStream，用于对象的反序列化

例如：打印流
（1）字节打印流：PrintStream
（2）字符打印流：PrintWriter

22、InputStream 里的 read()返回的是什么值,read(byte[] data)是什么意思,返回的是什么值。Reader 里的 read()返回的是什么值，read(char[] data)是什么意思，返回的是什么值。如果想要一次读取一行怎么办？

InputStream：
read()方法，返回的是所读取的字节的 int 型（范围 0-255）
read（byte[] data）将读取的字节储存在这个数组，返回的是实际读取的字节数。
Reader：
read()方法，返回的是所读取的字符的 int 型（范围 0-65535）
read(char[] data)将读取的字符存储在这个数组中，返回的是实际读取的字符数。
如何读取一行：
BufferedReader 类中有 readLine()方法。Scanner 类中也有 nextLine()方法。

23、Java 反射机制的作用？

反射就是动态加载对象，并对对象进行剖析。Java 反射机制的作用：
（1）在运行时创建任意类型的对象
（2）在运行时获取任意类型的信息
（3）在运行时获取和设置任意属性值
（4）在运行时调用任意对象的方法

24、如何获取 Class 的对象？4 种方式

获取 Class 对象的四种方式：
（1）类型名.class
（2）对象.getClass()
（3）Class.forName("类型的全名称")
（4）ClassLoader 对象.loadClass("类型的全名称")

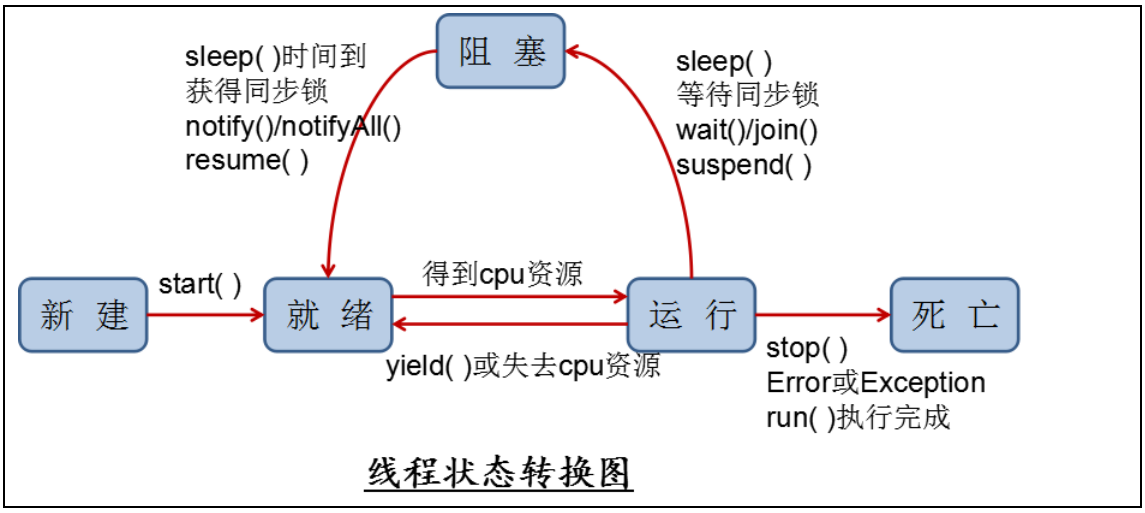
25、编写多线程程序有几种实现方式？

JavaSE 阶段考试答出两种即对：
1、继承 Thread 类，可以直接调用 start()启动，有单继承限制，共享数据时需要使用 static 方式，只能选择当前类.class 对象或其他共享对象当锁。
2、实现 Runnable 接口，必须借助 Thread 对象的 start()启动，实现接口可以解决单继承限制问题，需要共享数据时，共享同一个 Runnable 对象即可，线程安全锁可以直接选择 this 对象。

企业面试阶段答案一如下：
1、继承 Thread 类，可以直接调用 start()启动，有单继承的限制。
2、实现 Runnable 接口，必须借助 Thread 对象的 start()启动，实现接口可以解决单继承限制问题。
3、使用 ExecutorService、Callable、Future 实现有返回结果的多线程

企业面试阶段答案二如下：
Java 多线程实现方式主要有四种：继承 Thread 类、实现 Runnable 接口、实现 Callable 接口通过 FutureTask 包装器来创建 Thread 线程、使用 ExecutorService、Callable、Future 实现有返回结果的多线程。
其中前两种方式线程执行完后都没有返回值，后两种是带返回值的。

26、请阐述线程的生命周期？



线程状态转换图

27、Thread 的 start()和 Runnable 的 run()有什么区别？

Thread 的 start()：启动一个线程是调用 start()方法，使线程所代表的虚拟处理机处于可运行状态，这意味着它可以由 JVM 调度并执行。这并不意味着线程就会立即运行。
Runnable 的 run()：线程的线程体方法。所有线程类都必须实现的 run()方法。

28、sleep() 和 wait() 有什么区别？

sleep 和 wait 都会导致当前线程进入阻塞状态，被挂起。

sleep 不释放锁，睡眠时间到自动醒来，回到就绪状态
wait 是会释放锁，要通过 notify()或 notifyAll()唤醒，回到就绪状态

sleep 是在 Thread 类中声明的一个静态方法，Thread.sleep(毫秒)
wait 是在 Object 类中声明的非静态的方法，必须锁对象调用

29、请阐述什么是线程安全问题，如何解决？

JavaSE 阶段的答案：
当满足以下条件时，会出现线程安全问题：
（1）有多个线程
（2）使用共享数据
（3）有多句代码操作共享数据

如何解决？同步，即加锁

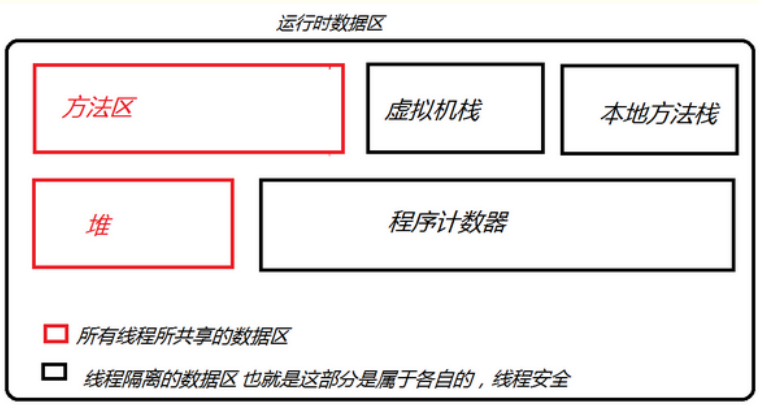
毕业的企业面试的答案，要学完 JUC 一起总结。

30、简要的写出进程和线程的区别（简单的写）？

（1）进程是操作系统资源的分配和调度的一个独立单元，而线程是 CPU 调度的基本单元
（2）同一个进程中可以包括多个线程，并且线程共享整个进程的资源（寄存器、堆栈、上下文），一个进程至少包括一个线程。

四、较难简答题（8 分/题）

1. Java 虚拟机中内存分为哪些区域？每个区域的作用？哪些区域是线程共享的？



1、程序计数器(寄存器)：当前线程所执行的字节码行号指示器
2、本地方法栈：同虚拟机栈，只不过本地方法栈为虚拟机使用到的 native 方法服务。
3、虚拟机栈：每个方法在执行的同时都会创建一个栈帧用来存放存储局部变量表、操作数表、动态连接、方法出口等信息，每一个方法从调用直至执行完成的过程，就对应着一个栈帧在虚拟机栈中入栈到出栈的过程。
4、堆：所有线程共享的一块内存区域。Java 虚拟机所管理的内存中最大的一块，因为该内存区域的唯一目的就是存放对象实例。几乎所有的对象实例度在这里分配内存，也就是通常我们说的 new 对象，同时堆也是垃圾收集器管理的主要区域。
5、方法区：和堆一样，是各个线程共享的内存区域，用于存储已被虚拟机加载的类信息、常量、静态变量、和编译器即时编译后的代码等

2. 请解释抽象类与接口的区别

JDK1.8 之前抽象类与接口的差别很大，JDK1.8 之后接口越来越像抽象类了。

		抽象类	接口
单继承限制		有	一个类可以实现多个接口， 而且接口也可以继承多个接口
成员	属性	可以有	只能有公共的静态的常量属性
	构造器	有	无
	代码块	可以有	无
	抽象方法	可以有	只能是公共的抽象方法
	静态方法	可以有	JDK1.8 之后可以有公共的静态方法
	方法的默认实现	可以有	JDK1.8 之后可以有公共的默认方法
相同点		都不能直接实例化，都是上层的抽象层	

3. Object 类中 equals 方法的实现是什么？重写一个 equals 方法有什么注意事项？

Object 类中的 equals 方法，对于任何非空引用值 x 和 y，当且仅当 x 和 y 引用同一个对象时，此方法才返回 true（x == y 具有值 true）。

在重写 equals 方法时，要注意满足离散数学上的特性

（1）自反性：对任意引用值 x，x.equals(x)的返回值一定为 true.

（2）对称性：对于任何引用值 x,y,当且仅当 y.equals(x)返回值为 true 时，x.equals(y)的返回值一定为 true;

（3）传递性：如果 x.equals(y)=true, y.equals(z)=true,则 x.equals(z)=true

（4）一致性：如果参与比较的对象没任何改变，则对象比较的结果也不应该有任何改变

（5）非空性：任何非空的引用值 x，x.equals(null)的返回值一定为 false

注意：当此方法被重写时，通常有必要重写 hashCode 方法，以维护 hashCode 方法的常规协定，该协定声明：

（1）相等对象必须具有相等的哈希码，

（2）两个对象的哈希码不相等，那么 equals 一定不相等。

两个对象的哈希码相等，那么 equals 结果可能相等也可能不相等

4. 比特(Bit),字节(Byte),字符(char/word),各有什么区别，通常说存储容量为 KB,MB,GB,TB 又是什么意思？通常说传输速率有 bps 和 Bps 有什么区别？

Bit 最小的二进制单位 ，是计算机的操作部分，取值 0 或者 1。

Byte 是是计算机信息技术用于计量存储容量的一种计量单位，由 8 位 bit 组成，取值（-128-127）。

char/word 是用户的可读写的最小单位，在 Java 里面一个 char 类型的变量占 2 个字节，取值（0-65535），但实际一个 char 存储到文件中占几个字节要看字符编码方式。

1KB = 1024Byte，1MB = 1024KB，1GB = 1024MB，1TB = 1024GB。
bps 是 bits per second 的简称，一般用于表示网络或 USB 等接口的数据传输速率。Bps 即是 Byte per second 的简称，电脑一般都以 Bps 显示速度，如 1Mbps 大约等同 128 KBps。

5. 运行时异常与编译时异常有何异同？请列举一些运行时异常和编译时异常的类型。

运行时异常是非受检异常，是 RuntimeException 的子类，即编译器无法检测，因此也不会强制要求程序员处理。 编译时异常是受检异常，编译器检测到代码抛出编译时异常时，会要求程序员必须对该异常做处理(throws 或 try...catch)否则，编译不通过。
运行时异常： 数组下标越界异常：ArrayIndexOutOfBoundsException 类型转换异常：ClassCastException 算术异常：ArithmeticException 空指针异常：NullPointerException 编译时异常： IO 操作异常：IOException 文件找不到异常：FileNotFoundException 已到达文件流末尾异常：EOFException 类找不到异常：ClassNotFoundException 没有对应的方法异常：NoSuchMethodException

6. HashMap 的底层实现及扩容机制？

简单回答： HashMap 在 JDK1.8 之前：底层实现是数组+链表，扩容机制是当 table 中元素的个数已经达到阈值（table.length*0.75）时并且新添加[index]桶已经是非空，那么 table.length 需要扩容为 2 倍。 HashMap 在 JDK1.8 之后：底层实现是数组+链表/红黑树，扩容机制（1）是当 table 中元素的个数已经达到阈值（table.length*0.75）时并且新添加[index]桶已经是非空，那么 table 需要扩容为 2 倍。（2）当添加到[index]下时，发现[index]下的链表结点个数已经达到 8 个，而 table 的长度未达到 64，此时 table.length 也会扩容为 2 倍

7.HashMap 的相关常量

DEFAULT_LOAD_FACTOR: 默认加载因子，值为 0.75 ***TREEIFY_THRESHOLD***：链表树化阈值，值为 8 ***MIN_TREEIFY_CAPACITY***：最小树化容量，值为 64

UNTREEIFY_THRESHOLD：反树化阈值，值为 6

8.如何实现序列化，有什么意义

如何实现序列化（5 分）： （1）实现 Serializable 接口或 Externalizable 接口，并且视情况而定指定一个序列化版本 ID（serialVersionUID）值；而且要保留公共的无参构造。 （2）如果某个对象的属性也是引用数据类型，那么该数据类型也要实现 Serializable 接口或 Externalizable 接口； （3）如果要序列化，则使用一个输出流来构造一个对象输出流 ObjectOutputStream 并通过 writeObject(Object obj)方法就可以将实现对象写出(即保存其状态)；如果需要反序列化则可以用一个输入流建立对象输入流 ObjectInputStream，然后通过 readObject 方法从流中读取对象。 （4）如果某些属性不参与序列化，如果是实现 Serializable 接口的，直接在属性前面加 transient 修饰，注意：static 修饰的属性也不会被序列化，如果是实现 Externalizable 接口，那么只要在重写 writeExternal()和 readExternal()方法时，不处理该属性即可。
意义（3 分）： 序列化就是一种用来处理对象流的机制，所谓对象流也就是将对象的内容进行流化，即把对象的内容转成二进制数据。可以对流化后的对象进行读写操作，也可将流化后的对象传输于网络之间。序列化是为了解决对象流读写操作时可能引发的问题（如果不进行序列化可能会存在数据乱序的问题）。

9.synchronized 关键字的用法？

synchronized 关键字是解决线程安全问题的方式之一。共有两种用法：
1、同步代码块
语法格式：
synchronized(锁对象){
　　需要加锁的代码
}
注意锁：
（1）任意类型的对象都可以当做锁
（2）多个线程之间共用一把锁，即多个线程之间共用同一个锁对象
（3）同步代码块的范围：不能太大，太小
 	
2、同步方法
语法结构：
synchronized 【修饰符】 返回值类型 方法名（【形参列表】）【抛出异常列表】
 	
同步方法的锁对象：
静态方法：当前类的 Class 对象，即当前类名.class
非静态方法：当前对象 this（需要谨慎，确保是同一个 this）

10.请列出你所知道的设计模式？

Java 中一般认为有 23 种设计模式，我们不需要所有的都会，但是其中常用的几种设计模式应该去掌握。总体来说设计模式分为三大类：

创建型模式，共五种：工厂方法模式、抽象工厂模式、单例模式、建造者模式、原型模式。

结构型模式，共七种：适配器模式、装饰器模式、代理模式、外观模式、桥接模式、组合模式、享元模式。

行为型模式，共十一种：策略模式、模板方法模式、观察者模式、迭代子模式、责任链模式、命令模式、备忘录模式、状态模式、访问者模式、中介者模式、解释器模式。

11.Object 中有哪些方法

（1）protected Object clone()--->创建并返回此对象的一个副本。
（2）boolean equals(Object obj)--->指示某个其他对象是否与此对象“相等”。
（3）protected void finalize()--->当垃圾回收器确定不存在对该对象的更多引用时，由对象的垃圾回收器调用此方法。
（4）Class<? extendsObject> getClass()--->返回一个对象的运行时类型。
（5）int hashCode()--->返回该对象的哈希码值。
（6）void notify()--->唤醒在此对象监视器上等待的单个线程。
（7）void notifyAll()--->唤醒在此对象监视器上等待的所有线程。
（8）String toString()--->返回该对象的字符串表示。
（9）void wait()--->导致当前的线程等待，直到其他线程调用此对象的 notify() 方法或 notifyAll() 方法。

void wait(long timeout)--->导致当前的线程等待，直到其他线程调用此对象的 notify() 方法或 notifyAll()方法，或者超过指定的时间量。

void wait(long timeout, int nanos)--->导致当前的线程等待，直到其他线程调用此对象的 notify()

12.请描述一下 JVM 加载 class 的过程和原理？

系统可能在第一次使用某个类时加载该类，但也可能采用预先加载机制来预加载某个类，不管怎样，类的加载必须由类加载器完成，系统会通过加载、连接、初始化三个步骤来对该类进行初始化。不管类的字节码内容从哪里加载，加载的结果都一样，这些字节码内容加载到内存后，都会将这些静态数据转换成方法区的运行时数据结构，然后生成一个代表这个类的 java.lang.Class 对象，作为方法区中类数据的访问入口（即引用地址），所有需要访问和使用类数据只能通过这个 Class 对象。

13、请阐述类加载器的类型

Java 的类加载器由如下四种：

1. 引导类加载器（Bootstrap Classloader）：又称为根类加载器

   它负责加载 Java 的核心库 JAVA_HOME/jre/lib/rt.jar 等，是用原生代码（C/C++）来实现的，并不继承自 java.lang.ClassLoder，所以通过 Java 代码获取引导类加载器对象将会得到 null。

2. 扩展类加载器（Extension ClassLoader）

   它由 sun.misc.Launcher$ExtClassLoader 实现，是 java.lang.ClassLoader 的子类，负责加载 Java 的扩展库 JAVA_HOME/jre/ext/*.jar 等。

3. 应用程序类加载器（Application Classloader）

   它由 sun.misc.Launcher$AppClassLoader 实现，是 java.lang.ClassLoader 的子类，负责加载 Java 应用程序类路径下的内容。

4. 自定义类加载器

   开发人员可以通过继承 java.lang.ClassLoader 类的方式实现自己的类加载器，以满足一些特殊的需求，例如对字节码进行加密来避免 class 文件被反编译，或者加载特殊目录下的字节码数据。

五、较难编程题（8 分/题）

1. 判断 101-200 之间有多少个素数，并输出所有素数

```
public static void main(String[] args) {
    System.out.println("101-200 之间的素数有：");
    for (int i = 101; i <= 200; i++) {
        boolean flag = true;
        for (int j = 2; j < i; j++) {
            if (i % j == 0) {
                flag = false;
                break;
            }
        }
        if (flag) {
            System.out.println(i);
        }
    }
}
```

2. 一个球从 100 米高度自由落下，每次落地后反跳回原高度的一半，再落下，求它在第 10 次落地时，共经过多少米？第 10 次反弹多高？

```
    public static void main(String[] args) {
        double height = 100;
        double distance = 0;
        int count = 10;
        for (int i = 1; i <= count; i++) {
            distance += height;// 加落下的距离
            height = height / 2;// 弹起的高度 第 i 次弹起的高度
            if (i != count) {
                distance += height; // 加弹起的距离
            }
        }
        System.out.println("第" + count + "次落地时，经过了：" + distance + "米");
        System.out.println("第" + count + "次反弹的高度是：" + height + "米");
    }
```

3. 用 100 元钱买 100 支笔，其中钢笔 3 元/支，圆珠笔 2 元/支，铅笔 0.5 元/支，问钢笔、圆珠笔和铅笔可以各买多少支？请写 main 方法打印需要买的数目。

```
    public static void main(String[] args) {
        double money = 100;
        double pPrice = 3;
        double yPrice = 2;
        double qPrice = 0.5;
        int count = 100;

        for (int pen = 1; pen <= money / pPrice; pen++) {
            for (int yuan = 1; yuan <= money / yPrice; yuan++) {
                for (int qian = 1; qian <= money / qPrice; qian++) {
                    if (pen + yuan + qian == count && pen * pPrice + yuan * yPrice +
qian * qPrice == money) {
                        System.out.println("购买" + pen + "支钢笔，" + yuan + "支圆珠
笔，" + qian + "支铅笔");
                    }
                }
```

```
            }
        }
    }
```

4. 通项公式如下：f(n)=n + (n-1) + (n-2) + ... + 1，其中 n 是大于等于 5 并且小于 10000 的整数，例如：f(5) = 5 + 4 + 3 + 2 + 1, f(10) = 10 + 9 + 8 + 7+ 6 + 5 + 4 + 3 + 2 + 1，请用递归的方式完成方法 long f(int n)的方法体。

```
    public static long f(int n) {
        long sum = 0;
        if(n==1){
            sum += 1;
        }else if(n>1){
            sum += n + f(n-1);
        }
        return sum;
    }
```

5. 求 1+2！+3！+...+20！的和

```
    public static void main(String[] args) {
        long sum = 0;
        for (int i = 1; i <= 20; i++) {
            sum += jieCheng(i);
        }
        System.out.println("sum = " + sum);
    }

    public static long jieCheng(int n){
        long temp = 1;
        for (int j = 1; j <=n; j++) {
            temp *= j;
        }
        return temp;
    }
```

6. 第一个人 10，第 2 个比第 1 个人大 2 岁，以此类推，请用递归方式计算出第 8 个人多大？

```
    public static void main(String[] args) {
        int count = 8;
        int age = getAge(count);
        System.out.println("第" + count +"个人的年龄：" + age);
    }

    public static int getAge(int count){
        if(count == 1){
            return 10;
        }else{
            return getAge(count-1) + 2;
        }
    }
```

7. 有 n 步台阶，一次只能上 1 步或 2 步，共有多少种走法？

答案一：递归

```
    public static int f(int n) {
        if (n <= 2)
            return n;
        int x = f(n - 1) + f(n - 2);
        return x;
    }
```

答案二：不用递归

```
    public static int f(int n) {
        if (n <= 2)
            return n;
        int first = 1, second = 2;
        int third = 0;
        for (int i = 3; i <= n; i++) {
            third = first + second;
            first = second;
            second = third;
        }
        return third;
    }
```

8. 输入整型数 98765，输出是 56789

```
    public static void main(String[] args) {
        long num = 98765L;
        StringBuffer sf = new StringBuffer(num + "");
        sf.reverse();
        String str = sf.toString();
        num = Long.parseLong(str);
        System.out.println(num);
    }
```

9. 有一个字符串，其中包含中文字符、英文字符和数字字符，请统计和打印出各个字符的字数。

举例说明： String content = “中中国 55kkfff”；
统计出：
中：2
国：1
5：2
k：2
f：3

```
    public static void main(String[] args) {
        String content = "中中国 55kkfff";
        HashMap<Character, Integer> map = new HashMap<Character, Integer>();
        while (content.length() > 0) {
            Character c = content.charAt(0);
            content = content.substring(1);
            Integer count = map.get(c);
            if (count == null) {
                map.put(c, 1);
            } else {
                map.put(c, count + 1);
            }
        }

        Set<Entry<Character, Integer>> entrySet = map.entrySet();
        for (Entry<Character, Integer> entry : entrySet) {
            System.out.println(entry);
        }
    }
```

10. 斐波纳契数列（Fibonacci Sequence），又称黄金分割数列。

一列数的规则如下：1、1、2、3、5、8、13、21、34....求第 n 位数是多少？
在数学上，斐波纳契数列以如下被以递归的方法定义：F0=0，F1=1，Fn=F(n-1)+F(n-2)（n>=2，n∈N*）在现代物理、准晶体结构、化学等领域，斐波纳契数列都有直接的应用

答案一：递归

```
public static long fibonacci(int n) {
    long result = 1;
    if (n > 2) {
        result = fibonacci(n - 1) + fibonacci(n - 2);
    }
    return result;
}
```

答案二：非递归

```
public static long fibonacci(int n) {
    long result = 1;
    if (n > 2) {
        long first = 1;
        long second = 1;
        int i = 0;
        n = n - 2;
        while (i < n) {
            first = second;
            second = result;
            result = first + second;
            i++;
        }
    }
    return result;
}
```

11. 请使用二分查找算法查找字符数组{“a”,”b”,”c”,”d”,”e”,”f”,”g”,”h”}中”g”元素的位置？

```
public static void main(String[] args) {
    char[] arr = { 'a', 'b', 'c', 'd', 'e', 'f', 'g', 'h' };

    char findValue = 'g';
    int findIndex = -1;
```

```
        int leftIndex = 0;// 最开始，左边的边界是 0
        int midIndex = arr.length / 2;// 最开始的中间：arr.length/2
        int rightIndex = arr.length - 1;// 最开始，右边的边界是 arr.length-1

        while (true) {
            if (arr[midIndex] == findValue) {
                findIndex = midIndex;
                break;
            } else {
                // 判断是否已经到达边界，如果是就结束查找过程
                // 如果不是，继续往左边或右边查找
                if (midIndex == 0 || midIndex == arr.length - 1) {
                    break;
                }

                // 判断是往左还是往右
                if (findValue < arr[midIndex]) {
                    // 往左边查找
                    rightIndex = midIndex;
                    midIndex = leftIndex + (rightIndex - leftIndex) / 2;
                } else {
                    // 往右边查找
                    leftIndex = midIndex;
                    midIndex = leftIndex + (rightIndex + 1 - leftIndex) / 2;
                }
            }
        }

        if (findIndex == -1) {
            System.out.println(findValue + "在数组中不存在");
        } else {
            System.out.println(findValue + "在数组中的位置就是" + findIndex);
        }
    }
```

12. 消除下面集合中重复元素？

List list = Arrays.asList(1,2,3,3,4,4,5,5,6,1,9,3,25,4);

```
import java.util.Arrays;
import java.util.HashSet;
import java.util.List;
```

```java
public class Test {
    public static void main(String[] args) {
        List<Integer> list = Arrays.asList(1, 2, 3, 3, 4, 4, 5, 5, 6, 1, 9, 3, 25, 4);
        HashSet<Integer> set = new HashSet<Integer>();
        set.addAll(list);

        for (Integer integer : set) {
            System.out.println(integer);
        }
    }
}
```

13. 请用 wait()和 notify()方法编写一个生产者消费者设计模式程序?

```java
import java.util.Random;

public class Test {
    public static void main(String[] args) {
        Houseware h = new Houseware();

        Worker w = new Worker(h);
        Saler s = new Saler(h);

        w.start();
        s.start();
    }
}

class Houseware {
    private Object[] buffer = new Object[10];
    private int total;

    synchronized public void put(Object data) {
        if (total >= buffer.length) {
            try {
                this.wait();
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
        }
```

```
        buffer[total] = data;
        total++;
        System.out.println(data + "被存入，现在数量是：" + total);
        this.notify();
    }

    synchronized public Object take() {
        if (total <= 0) {
            try {
                this.wait();
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
        }
        Object data = buffer[0];
        System.arraycopy(buffer, 1, buffer, 0, total - 1);
        total--;
        this.notify();
        System.out.println(data + "被取出，现在数量是：" + total);
        return data;
    }
}

class Worker extends Thread {
    private Houseware h;

    public Worker(Houseware h) {
        super();
        this.h = h;
    }

    public void run() {
        Random r = new Random();
        while (true) {
            h.put(r.nextInt());
        }
    }
}

class Saler extends Thread {
    private Houseware h;

    public Saler(Houseware h) {
        super();
```

```
        this.h = h;
    }

    public void run() {
        while (true) {
            Object take = h.take();
        }
    }
}
```

六、附加题（10 分）

1、编写代码完成如下功能
public static String replace(String text, String target, String replace){
    ....
}
示例：replace(“aabbccbb”, “bb”, “dd”); 结果：aadccdd
注意：不能使用 String 及 StringBuffer 等类的 replace 等现成的替换 API 方法。

2、1 个字符串中可能包含 a-z 中的多个字符，字符也可能重复，例如：String data = “aabcexmkduyruieiopxzkkkkasdfjxjdsds”;写一个程序，对于给定一个这样的字符串求出字符串出现次数最多的那个字母以及出现的次数（若次数最多的字母有多个，则全部求出）

3、假设日期段用两个 6 位长度的正整数表示，例如：(201401，201406)用来表示 2014 年 1 月到 2014 年 6 月，求两个日期段的重叠月份数。例如：输入：201401 和 201406，201403 和 201409，输出：4
解释：重叠月份：3,4,5,6 月共 4 个月

4、入参为一个整型数组（Integer[] input），要求对入参(input)按奇偶数分成两个数组，要求启动两个线程，分别将入参(input)中的奇数和偶数输出到一个文件中，需要偶数线程每打印 10 个偶数以后，就将奇数线程打印 10 个奇数，如此交替进行。同时需要记录输出进度，每完成 1000 个数就在控制台中打印当前完成数量，并在所有线程结束后，在控制台打印“Done”

5、编程实现单向链表，并实现单向链表的反转。比如一个链表是这样的：1->2->3->4->5，通过反转后成为 5->4->3->2->1，注：即实现单向链表类，在该类中提供一个单向链表的反转方法 reverse，请写出完整代码

6、找出数组中一个值，使其左侧值的加和等于右侧值的加和，例如：1,2,5,3,2,4,2，结果为：第 4 个值

7、编程实现：线程 A 向队列 Q 中不停写入数据，线程 B 从队列 Q 中不停读取数据（只要 Q

中有数据）

8、写一个排序算法 1-100 随机数字，进行排序，要求效率（例如：冒泡、选择、快排.....等）