# 杰普大数据入学测试

**一、选择题（每题 2 分，共 30 题）**

1、欲构造 ArrayList 类的一个实例，此类继承了 List 接口，下列哪个方法是正确的?

A、ArrayList myList = new Object();

B、List myList = new ArrayList();

C、ArrayList myList = new List();

D、List myList = new List();

2、下面的代码段中，执行之后 i 和 j 的值是什么?

int i = 1; int j;

j = i++*2+3*--i;

A、1, 2 B、1, 5 C、 2, 1 D、 2, 2

3、下面哪条语句把方法声明为抽象的公共方法?

A、public abstract method();

B、public abstract void method();

C、public abstract void method(){}

D、public void method() extends abstract;

4、下面哪个语句（初始化数组）是不正确的?

A、int x[] = {1,2,3}; B、int x[3] = {1,2,3};

C、int[] x = {1,2,3}; D、int x[] = new int[]{1,2,3};

5、如果要在字符串类型对象 s = ”JAVA”；中找出”V”出现的位置可使用下面哪个方法?

A、mid(2,s); B、charAt(2); C、s.indexOf(“V”); D、indexOf(s, “V”);

6、类 A 派生出子类 B, B 派生出子类 C，并且在 Java 源代码中有如下声明：

1) A a0=new A(); 2) A a1 =new B(); 3) A a2=new C();

问以下哪个说法是正确的?

A、只有第 1 行能通过编译

B、第 1、2 行能通过编译，但第 3 行编译出错

C、第 1、2、3 行能通过编译，但第 2、3 行运行时出错

D、第 1 行、第 2 行和第 3 行的声明都是正确的

7、下列说法错误的有?

A、数组是一种对象 B、数组属于一种原生类

C、int number[] = {31,23,33,43,35,63} D、数组的大小可以任意改变

8、下列标识符不合法的有?

A、new B、$Usdollars C、1234 D、car.taxi

9、 以下哪些描述是正确的?

A、int 是 32 位的带符号型整型值

B、short 型是 16 位的带符号整形值

C、char 型是 16 位的 Unicode 编码字符

D、float 型是 64 位的浮点值

10、以下哪些是逻辑短路运算符?

A、& B、&& C、| D、||

11、现有一组 int MyArray[]={1,2,3} 以下哪些语句用于获取该数组的元素个数?

A、MyArray.size; B、MyArray.size(); C、MyArray.length;

D、MyArray.length();　　E、MyArray.getSize();

12、以下哪些语句用于声明一个二维数组?

A、int a[][]=new int[10,10];　　B、int a[][]=new int [10][10];

C、int a[10,10]=new int[10][10];　　D、int [][]a=new int [10][10];

13、如何强制垃圾回收器回收一个对象?

A、调用 System.gc()方法　　B、调用 Runtime.gc()方法

C、将对象赋值 null　　D、无法强制垃圾回收器执行

14、下面代码执行结果是?

```
class Test{
    int num = 81;
    public static void main(String[] args){
        new Test().go();
    }
    void go(){
        incr(++num);
        System.out.println(num);
    }
    void incr(int num){    num+=10; }
}
```

A、81　B、82　C、91　D、92　E、编译错误　F、运行时，抛出异常

15、在 Java 中下面关于构造方法说法错误的是?

A、对于每一个类，Java 虚拟机都提供一个默认构造方法

B、构造方法不能够被重载

C、构造方法可以接受参数

D、当类的父类只有一个带参数的构造方法时，这个类必须提供自定义的构造函数

16、设 String s = "story"; 下列选项中的语句书写正确的是?

A、s += "books";　　B、char c = s[1];

C、int len = s .length;　　D、s = s - "books";

17、当下列程序执行时，其输出结果是?

```
int x = 5;
int y = 2;
System.out.println(x+ y + "k");
```

A、5k　　B、7k　　C、52k　　D、3k

18、以下对抽象类的描述正确的是?

A、抽象类没有构造方法

B、抽象类必须提供抽象方法

C、有抽象方法的类一定是抽象类

D、抽象类可以通过 new 关键字直接实例化

19、在 JAVA 中，什么关键字用来终止循环语句?

A、return　　B、continue　　C、break　　D、exit(0)

20、下面代码执行结果是?

```
class A {
    public String show(D obj) { return ("A and D"); }
    public String show(A obj) { return ("A and A"); }
}
class B extends A {
    public String show(B obj) { return ("B and B"); }
```

```
        public String show(A obj) { return ("B and A"); }
    }
    class D{}
    public class Test{
        public static void main(String[] args){
            A a = new B();
            B b = new B();
            System.out.print(a.show(b));
        }
    }
```

A、A and D　　B、A and A　　C、B and B　　D、B and A

21、给定代码片段，会出错的是?

```
(1)    String str  =  null;
(2)    if( ( str  !=  null )  &&  ( str.length() > 10 ) ){
(3)            System.out.println(“more than 10”);
(4)    } else if( ( str  !=  null) & ( str.length( ) < 5 ) ){
(5)            System.out.println(“less than 10”);
(6)    } else {
(7)            System.out.println(“end”);
(8)    }
```

A、第 1 行　　B、第 2 行　　C、第 4 行　　D、第 8 行　　E、无错误

22、执行上面代码，结果是?

```
interface Door{}
class WoodDoor implements Door{}
class AlarmDoor extends WoodDoor{}
class Window{}
public class DoorApp{
    public static void main(String[] args){
        Door d = new WoodDoor ();
        AlarmDoor a = new AlarmDoor();
        Window w = new Window();
        if(d instanceOf WoodDoor){ System.out.print(“d-w ”);}
        if(a instanceOf Door){ System.out.print(“a-d”);}
        if(w instanceOf Door){ System.out.print(“w-d”);}
    }
}
```

A、d-w　　B、d-w a-d　　C、d-w w-d　　D、d-w a-d w-d

E、编译错误　F、运行时，抛出异常

23、设x为float型变量, y为double型变量, a为int型变量, b为long型变量, c为char型变量，则表达式 x+y*a/x+b/y+c 的值为什么类型?

A、int　　B、long　　C、double　　D、char

24、给定如下 java 代码，编译运行之后，将会输出?

```
public class Test{
    public static void main(String args[]){
        int a=5;
        System.out.println((a%2==1) ?(a+1) /2:a/2) ;
    }
}
```

A、1　　B、2　　C、2.5　　D、3

25、在 switch(expression)语句中, expression 的数据类型不能是?

A、double　　B、char　　C、byte　　D、short

26、执行下列代，打印结果是?

```
Integer a1=128,a2=128;
String s1="abc",s2="abc";
Double d1=12.0,d2=12.0;
Byte b1=12,b2=12;
System.out.println(a1==a2);
System.out.println(s1==s2);
System.out.println(d1==d2);
System.out.println(b1==b2);
```

A、true,true,true,true　　B、false,true,false,true　　C、false,true,false,false

D、true,false,true,false　　E、true,false,false,false　　F、false,true,true,true

27、下面程序的运行结果是?

```
String str1 = "hello";
String str2 = "he" + new String("llo");
System.err.println(str1 == str2);
System.err.println(str1.hashCode() == str2.hashCode());
```

A、true,true　　B、true,false　　C、false,true　　D、false,false

28、两段代码中，list 空间分别扩充了几次?

**(1)**ArrayList list = new ArrayList <Integer> (20);

**(2)**ArrayList list = new ArrayList <Integer> ();

```
for(int i=0;i<20;i++)
    list.add(i);
```

A、0 次和 2 次　　B、1 次和 0 次　　C、0 次和 20 次　　D、20 次和 1 次

29、给出以下代码，请问变量 a 取何值，该程序打印输出 if-2?

```
if(a>4) System.out.println("if-1");
else if(a>9) System.out.println("if-2");
else System.out.println("if-3");
```

A、10　　B、1　　C、-5　　D、无论取值均不可能打印输出 if-2

30、以下哪些表达式的结果是合法的?

A、int x=6;x=!x　　B、int x=6;if(!(x>3)){}

C、int x=6;x=~x;　　D、int x=1;x=x^3;

**二、填空题（每题 1 分，共 10 题）**

1、面向对象程序设计的三个特征是_____, _____, _____.

2、Java 是面向对象语言，类是客观事物的____，而对象是类的____.

3、Java 实现多态的机制是_____和_____.

4、Java 中为了克服____ 的缺点，使用了接口，一个类可以实现多个接口.

5、Java 使用____字符集.

6、当把级别高的变量的值赋予级别低的变量时，必须使用 ____ 转换。

7、在 Java 中，存在使 i + 1 < i 成立的数吗?请写出来____.

8、请写出一个使 i > j || i <= j 不成立的数。_____.

9、类加载机制是_____.

10、在长度为 n 的有序线性表中进行二分查找，最坏的情况下，需要的比较次数为____.

## 三、数据库试题（每空 2 分，共 10 分）

某工程项目公司的信息管理系统部分关系模式如下：
职工 Employee(编号 ecode，姓名 ename，性别 egender，居住城市 ecity)
项目 Project(项目编号 pcode，项目名称 pname，状态 pstate，城市 pcity，负责人编号 ecode)
职工项目 EmpPro（职工编号 ecode，项目编号 pcode）

一个职工可以同时参与多个项目，一个项目需要多个职工参与
每个项目必须有负责人，且负责人为职工关系中的成员
项目状态有两个：0 表示未完成，1 表示已完成
问题填空：
1 查询至少参加两个项目的职工编号和参与的项目数

```
select ecode, _______A_________
from EmpPro
group by _________B_____________
having _____________C_______________
```

2 查询参与居住城市正在进行的工程项目的职工工号和姓名

```
select E.ecode, E.ename
from Employee    E , EmpPro    EP, Project    P
where E.ecode = EP.ecode
and P.pcode = EP.pcode
and _____________D_______________
and_____________E_______________
```

## 四、程序填空（每空 5 分，共 20 分）

1、“背包问题”的基本描述是: 有一个背包, 能盛放的物品总重量为 totalWeight, 设有 N 件物品, 其重量分别为 w1, w2, ..., wn, 希望从 N 件物品中选择若干件物品, 所选物品的重量之和恰能放入该背包, 即所选物品的重量之和等于 totalWeight。
如下程序均能求得“背包问题”的一组解(采用递归算法), 请完善程序

```
public class BagTest {
  private int[] gw;
  //若干个物品的重量值
  public BasicTest(int[] gw){    this.gw = gw;}
  //装入包中物品的总重量为weight
  public boolean knap(int weight){   return knap(weight, gw.length - 1);}
  private boolean knap(int s, int n){
      if(s == 0)return true;
      if(s < 0 || ( s > 0 && n < 1)) return false;
      if( ___________A___________){
          System.out.printf("%4d\n", gw[n]);
          return true;
      }
      Return ___________B___________;
```

```
    }
    public static void main(String[] args) {
      int totalWeight = 15;
      //装包的总重量
      int[]  w=new int[]{0,1,4,3,4,5,2,7};
      //若干个物品的重量值
      BasicTest bt = new BasicTest(w);
      if(bt.knap(totalWeight)) System.out.println("OK!");
      else System.out.println("NO!");
    }
}
```

2、按索引访问集合，集合(List)的实现分为两种，一是基于数组，二是基于链表，基于链表的 ArrayList 适合数据查询操作，而基于链表的 LinkedList 则更加适合添加、删除操作。以下是两种存储结构的集合(List)的不同实现。看懂两种算法，并完成程序填空。

**a)基于数组的算法分析**

使用一个数组来描述集合，需要把集合中的每个元素映射到数组的具体位置上。下面我们用一个数学公式来确定每个元素的位置。一个简单的映射公式如下：

location(i)=i-1 (2-1)

公式(2-1)指明集合中第 i 个元素（如果存在的话）位于数组中 i-1 位置处。为了完整的描述集合，使用变量 length 作为集合的长度。

在这种数据结构中，搜索一个元素只需根据公式(2-1)进行映射就能实现，其平均时间复杂度是 O (1)，性能非常理想。

**元素删除**: 为了在集合中删除第 k 个元素，需要首先确认集合中包含了第 k 个元素（如果不存在第 k 个元素，则引发一个异常），将元素 k+1，k+2，…，length 依次向前移动一个位置，并将 length 的值减 1，从而删除第 k 个元素。删除操作的平均时间复杂度为 O(length)。

**插入元素**: 为了在集合中第 k 个元素后面插入一个新元素，首先要把 k+1 至 length 元素往后移动一个位置，然后把新元素插入到 k+1 位置处。插入操作的平均时间复杂度为 O(length)。此外，插入操作是，如果数组已经满了，则会引发一个异常。

采用数组来描述一个集合实现起来非常简单。执行搜索的平均时间复杂度为常数，非常理想。执行删除和插入的时候，有一个和集合的大小呈线性关系的平均时间复杂度，在大部分情况下也是可以接受的。利用数组来描述集合，使用在删除和插入操作少，搜索操作多的场合有非常优异的表现。

但是，这种描述方法有一个非常大的缺点是空间的低效利用。必须要预测集合的最大可能尺寸，根据最大可能尺寸分配数组。考察如下情况，我们需要一个集合，它的最大可能尺寸是 5000，但平均尺寸只有 100，那么空间的利用率就只有 2％。在实际应用中，可能发生的情况会比这更极端。

**b)基于单链表(以下称为链表)的算法分析**

在链表描述中，集合中的元素都放在链表的节点中进行描述。链表中的节点不是一个数组元素，因此不能通过公式来映射定位某个元素。取而代之的是，每个节点中都包含了下一个节点的位置信息，链表的表头包含了第一个节点的位置信息。

**索引元素**: 为了在集合中找到第 k 个元素，就必须从表头开始，遍历第 1 个到第 k 个节点。它的时间复杂度是 O(k)，平均时间复杂度为 O(length)。

**元素删除**: 为了在集合中删除第 k 个元素，就要先找到第 k-1 和第 k 个节点，使第 k-1 个节点的下一个节点位置信息指向第 k+1 个节点，然后释放第 k 个节点所占的空间。它的时

间复杂度是 O(k)，平均时间复杂度为 O(length)。

**插入元素**: 插入和删除的过程很相似，首先要创建一个新的节点，然后找到第 k-1 个节点，在该节点的后面插入新的节点，并把第 k-1 个节点、新的节点的下一个节点位置信息进行适当设置。它的时间复杂度是 O(k)，平均时间复杂度为 O(length)。

采用数组描述方法的集合仅需要能够保存所有元素的空间以及保存集合最大尺寸所需要的空间。链表描述需要除集合元素本身的空间意外，还需要额外的空间，用例保存链接节点指向下一个节点位置的指针。但一般情况下，链表描述要比数值描述的空间利用率高得多。

虽然数组描述、链表描述插入和删除操作的平均时间复杂度均为 O(length)，但由于移动元素的操作比遍历元素的操作的开销要大得多，所以采用链表描述所实现的插入和删除操作要比数组描述执行得更快。而采用数组描述可以在常数时间内访问第 k 个元素，而在链表中，这种操作所需要的时间是 O(k)。

**List 接口：**

```
public interface List {
  void insert(int index, Object obj)throws Exception;
  Object remove(int index)throws Exception;
  Object get(int index)throws Exception;
}
```

**基于数组的集合实现：**

```
public class ArrayList implements List {
  private int count = 0;
  private Object[] elements;
  private static final int INCREMENT = 10;
  public ArrayList(int size){ elements = new Object[size];}
  public void insert(int index, Object obj) throws Exception {
       encapacity();
       if(index > elements.length)throw new Exception("");
       for(int i = elements.length; i > index; i--)
             (__________A__________)
       elements[index] = obj;
       count ++;
  }
  private void encapacity() {
       if(count > elements.length){
            Object[] ns = new Object[elements.length + INCREMENT];
            System.arraycopy(elements, 0, ns, 0, elements.length);
            elements = ns;
       }
  }
  public Object remove(int index)throws Exception{
       if(index >= elements.length)throw new Exception("");
       Object o = elements[index];
       for(int i = index; i < elements.length - 1; i++)
            (__________B__________);
```

```
        count--;
        return o;
    }
    public Object get(int index) throws Exception {
        if(index >= elements.length)throw new Exception("");
        return elements[index];
    }
}
```

**基于链表的集合实现：**

```
public class LinkedList implements List {
    private class Node{
        Object   elements;
        Node     next;
    }
    private Node      head;
    private int       count = 0;
    public LinkedList(){ head = new Node();}
    public void insert(int index, Object obj) throws Exception {
        if(index > count)throw new Exception("");
        Node temp = indexAt(head, index);
        Node nn = new Node();
        (_____C_____);
        (_____D_____);
        count ++;
    }
    private Node indexAt(Node node, int index) {
        while(index > 0)node = node.next;
        return node;
    }
    public Object remove(int index) throws Exception {
        if(index >= count)throw new Exception("");
        Node previous = indexAt(head, index);
        Object r = previous.next.elements;
        (_____E_____);
        return r;
    }
    public Object get(int index) throws Exception {
        if(index >= count)throw new Exception("");
        Node previous = indexAt(head, index);
        return (_____F_____);
    }
}
```

3、快速排序（Quicksort）是对冒泡排序的一种改进，它的基本思想是：通过一趟排序将要排序的数据分割成独立的两部分，其中一部分的所有数据都比另外一部分的所有数据都要小，然后再按此方法对这两部分数据分别进行快速排序，整个排序过程可以递归进行，以此达到整个数据变成有序序列。

设要排序的数组是 A[0]……A[N-1]，首先任意选取一个数据（通常选用数组的第一个数）作为关键数据，然后将所有比它小的数都放到它前面，所有比它大的数都放到它后面，这个过程称为一趟快速排序。值得注意的是，快速排序不是一种稳定的排序算法，也就是说，多个相同的值的相对位置也许会在算法结束时产生变动。

一趟快速排序的算法是：

1）设置两个变量 i、j，排序开始的时候：i=0, j=N-1;

2）以第一个数组元素作为关键数据，赋值给 key，即 key=A[0];

3）从 j 开始向前搜索，即由后开始向前搜索(j--)，找到第一个小于 key 的值 A[j]，将 A[j]和 A[i]互换;

4）从 i 开始向后搜索，即由前开始向后搜索(i++)，找到第一个大于 key 的 A[i]，将 A[i]和 A[j]互换;

5）重复第 3、4 步，直到 i=j；（3,4 步中，没找到符合条件的值，即 3 中 A[j]不小于 key，4 中 A[i]不大于 key 的时候改变 j、i 的值，使得 j=j-1，i=i+1，直至找到为止。找到符合条件的值，进行交换的时候 i，j 指针位置不变。另外，i==j 这一过程一定正好是 i+或 j-完成的时候，此时令循环结束）.

```
public class Quicksort<T extends Comparable<? super T>> {
  public  void quickSort(T[] array,int start,int end){
       if(end<=start)return ;
       if(end-start==1){
            if(array[start].compareTo(array[end])>0){
                 T t=array[start];
                 array[start]=array[end];
                 array[end]=t;
            }
            return ;
       }
       int i=start;
       int j=end;
       int k=i;
       while(i<j){
            while(i<j){
                 if(array[k].compareTo(array[j])>0){
                      T t=array[k];
                      array[k]=array[j];
                      array[j]=t;
                      k=j;
                      i++;
                      break;
                 }
                 j--;
            }
            while(i<j){
                 if(array[i].compareTo(array[k])>0){
                      T t=array[i];
                      array[i]=array[k];
                      array[k]=t;
                      k=i;
                      j--;
                      break;
                 }
                 (__________A__________);
            }
       }
       (__________B__________);
       quickSort(array,k+1,end);
  }
  public static void main(String[] args) {
       Integer a[] = {6,2,7,3,8,9,5};
       for(Integer i:a)
            System.out.print(i+" ");
       new Quicksort<Integer>().quickSort(a, 0, a.length-1);
       System.out.println();
       for(Integer i:a)
            System.out.print(i+" ");
  }
}
```