算法设计考试说明（20170605）

1.作弊者无论前面通过几次，分数一律为 0

2.每次考试时间 30 分钟，没有在规定时间上交者，后果自负（此次分数为 0）

3.没有按要求的代码输入、输出方式编码者分数一律为 0

4.缺考者没有补考机会，缺者此次考试分数为 0

5.严格按要求的时间考试，未按安排考试时间考试者不得参与后续场次考试

6.考试必须带考试证件（身份证和学生证或学生卡），否则严禁进入考试，后果自负

7.一个学生只能登录一次，退出后禁止登录

8.再次强调：上交时不要拷贝结果，只粘贴源代码，以后拷贝结果的不用再写查分申请，一律按错误处理

9.考试事项解释权

《算法设计与分析》第一次章节考核时间安排

班级		时间

第一次单元章节考核

要求：

（1）要求动态通过键盘在对话框或命令行输入，而后显示结果（输入输出用 scanner 或 system.in），**本次考试的输入不要采用 JOptionPane 的输入方式**；
（2）程序检查过后点“提交”按钮，然后关机，走人！

注意：

（1）严格按照输入、输出的要求，否则一律 0 分，补考！！！

题目如下：

题目如下：

1. Ackerman 函数的**递归实现**算法。
输入：输入两个数字，先输入 n，后输入 m。
输出：Ackerman 函数计算后的值。
示例：输入：4　　2，输出：16

Ackerman 函数 A(n，m)定义如下：

有两个独立的整型变量 m、n：

$$\begin{cases} A(1,0)=2 & m=0,n=1 \\ A(0,m)=1 & m\geq 0,n=0 \\ A(n,0)=n+2 & m=0,n\geq 2 \\ A(n,m)=A(A(n-1,m),m-1) & m\geq 1,n\geq 1 \end{cases}$$

2.全排列的**递归实现**算法。

输入：先输入要求输入的字符的个数，后依次输入（或随机生成）每个字符（不能仅仅是数字）。

输出：全排列的结果。

示例：输入：3 / * 2，输出：/ * 2 / 2 * * / 2 * 2 / 2 * / 2 / *

R 的全排列可归纳递归定义如下：

当n=1时，perm(R)=(r)，其中r是集合R中唯一的元素；
当n>1时，perm(R)由(r_1)perm(R_1)，(r_2)perm(R_2)，...，(r_n)perm(R_n)构成。

3.整数划分的递归实现算法。

输入：输入整数划分的整数（只输入一次，即 n==m）。

输出：输入整数的划分个数值。

示例：输入：7，输出：15

q(n,m)的如下递归关系定义如下：

正整数 n 的划分数 p(n)=q(n,n)。

$$q(n,m)=\begin{cases} 1 & n=1,m=1 \\ q(n,n) & n<m \\ 1+q(n,n-1) & n=m \\ q(n,m-1)+q(n-m,m) & n>m>1 \end{cases}$$

4.二分搜索的**递归实现**算法。

输入：先输入进行二分搜索元素的个数，然后按大小依次输入（或随机生成，然后排序）每个数字，最后输入要求搜索的元素。

输出：要求搜索元素的下标（下标从 0 开始！）。

示例：输入：6 1 5 5 9 6 9 6，输出 3

5.合并排序的**递归实现**算法。

输入：先输入进行合并排序元素的个数，然后依次随机输入（或随机生成）每个数字。

输出：元素排序后的结果，数字之间不加任何标识符。
示例：输入：8　11　1　2　4　8　6　15　8，输出：1　2　4　6　8　8　11　15

6.快速排序的**递归实现**算法。
输入：先输入进行合并排序元素的个数，然后依次随机输入（或随机生成）每个数字。
输出：元素排序后的结果。
示例：输入：8　9　1　2　4　8　6　15　8，输出：1　2　4　6　8　8　9　15

第二次单元章节考核

要求：

（1）要求动态通过键盘在对话框或命令行输入，而后显示结果（输入输出用 scanner 或 system.in），**本次考试的输入不要采用 JOptionPane 的输入方式；**
（2）程序检查过后点“提交”按钮，然后关机，走人！

注意：

（1）严格按照输入、输出的要求，否则一律 0 分，补考!!!

题目如下：

1.写出菲波拉契数列自底向上的非递归动态规划算法或自顶向下的递归动态规划算法（备忘录方法）。
输入：输入一个数字。
输出：输出为 Fibonacci 数列的值。
示例：输入：5　，输出：8

Fibonacci 数列可以递归地定义为：

$$F(n)=\begin{cases}1 & n=0\\ 1 & n=1\\ F(n-1)+F(n-2) & n>1\end{cases}$$

2. 写出矩阵连乘的自底向上非递归的动态规划算法或自顶向下递归的动态规划算法（备忘录方法）。
输入：先输入矩阵连乘的个数 n，然后依次手动输入（不能随机生成!）矩阵的维数 pi（数字）。注意，6 个矩阵，需输 7 个维数值。
输出：矩阵连乘的次序，如：((A1(A2A3))((A4A5A6))。
示例：输入：6　30　35　15　5　10　20　25，输出：((A1(A2A3))((A4A5)A6))

因为k有$j-i$种选择，即$k=i$，$i+1$，…，$j-1$。最优加括号方式一定利用某个k值，我们只需逐个检查，找出最优的。因此，矩阵乘积$A_iA_{i+1}\ldots A_j$加括号的最小开销的递归定义变为

$$m[i,j]=\begin{cases}0 & i=j\\ \min\limits_{i\le k<j}\{m[i,k]+m[k+1,j]+p_{i-1}p_kp_j\} & i<j\end{cases} \qquad (3.4)$$

$m[i,j]$给出了子问题最优解的值。为了记录构造最优解的过程，设$s[i,j]$表示将$A_iA_{i+1}\ldots A_j$分裂产生最优解时k的位置，即$s[i,j]$等于值k，满足$m[i,j]=m[i,k]+m[k+1,j]+p_{i-1}p_kp_j$。

3. 写出 0-1 背包问题的自底向上非递归的动态规划算法。

输入：首先输入物品的个数 n，然后输入背包的容量 c，再依次输入每个物品的重量 wi，最后依次输入每个物品的价值 vi。注意：所有值都不能随机生成!!!

输出：物品的选择向量。如：(1,0,0,1,1)等。

示例：输入：4 5 2 1 3 2 12 10 20 15 输出：1 1 0 1 或（1,1,0,1）

由问题的最优子结构性质，我们可以建立计算**m(i, j)**的递归式如下：

$$m(i,j)=\begin{cases}\max\{m(i+1,j),m(i+1,j-w_i)+v_i\} & j\ge w_i\\ m(i+1,j) & 0\le j<w_i\end{cases}$$

$$m(n,j)=\begin{cases}v_n & j\ge w_n\\ 0 & 0\le j<w_n\end{cases}$$

4. 写出最优二叉搜索树的自底向上非递归的动态规划算法。

输入：首先输入结点的个数 n，再依次输入搜索成功的概率 bi，最后依次输入搜索失败的概率 aj。注意：所有值都不能随机生成，且只输入整数（概率×100）!!!

输出：最优二叉树的结构。

示例：输入：5 15 10 5 10 20 5 10 5 5 5 10，输出：

s2是根

s2的左孩子是s1

s2的右孩子是s5

s5是根

s5的左孩子是s4

s4是根

s4 的左孩子是 s3

递归方程(3.14)中假设我们知道结点s_k作为根结点。如果没有这个假设，我们需要选择具有最小开销的那个结点作为根，则可得出以下递归方程：

$$r[i,j]=\begin{cases} q_{i-1} & j=i-1 \\ \min\limits_{i\le r\le j}\{r[i,k-1]+r[k+1,j]+w(i,j)\} & i\le j \end{cases}$$

$r[i, j]$给出了最优二分检索树的期望开销。为了记录最优二分检索树的生成过程，我们定义root$[i, j]$为下标k，$1\le i\le j$，这个k使得s_k成为关键字s_i，…，s_j构成的最优二分检索树的根。

$$w[i,j]=w[i,j-1]+p_j+q_j \qquad (3.16)$$

因此，计算了$\Theta(n^2)$个$w[i, j]$值，每个$w[i, j]$的计算时间为$\Theta(1)$。

第三次单元章节考核

要求：

（1）要求动态通过键盘在对话框或命令行输入，而后显示结果（输入输出用 scanner 或 system.in），**本次考试的输入不要采用 JOptionPane 的输入方式**；
（2）程序检查过后点“提交”按钮，然后关机，走人！

注意：

（1）严格按照输入、输出的要求，否则一律 0 分，补考!!!

题目如下：

1. 写出活动安排问题的贪心算法。
输入：先输入活动的个数 n，然后依次输入每个活动的开始时间 si 及结束时间 fi。如：时间输入格式为：8：30 输入为 830。注意：所有值都不能随机生成!!!
输出：活动的选择向量，即 0，1 的集合。如： 1 0 1 0 1 或（1，0，1，0，1）
示例：输入：9 800 1030 900 1130 700 1100 1130 1400 1200 1330 1300 1530 1500 1600 1430 1600 1600 1800 ，输出：1 0 0 0 1 0 1 0 1

（原始的答案是没有排序的结果）

2. 写出一般背包问题的贪心算法。
输入：先输入物品的个数 n，再输入背包的容量 c，然后依次输入物品的重量 wi，最后依次输入物品的价值 vi。注意：所有值都不能随机生成!!!
输出：物品的选择向量 xi，0=<xi=<1。
示例：输入：3 20 18 15 10 25 24 15，输出：0.0 1.0 0.5

（原始的测试案例重量 价值弄反了）

```
public static void main(String[] args)
{
    Scanner sc = new Scanner(System.in);
    System.out.println("请输入个数");
    int n = sc.nextInt();
    int arrey[] = new int[n];
    for (int i = 0; i < arrey.length; i++)
    {
        arrey[i] = sc.nextInt();
    }

    /**
     * nextInt()读取int
     * nextDouble()读取double
     * 注意循环中不要使用JOptionPane来读取
     * Scanner最好只定义一次，使用的时候只需调用nextInt或nextDouble即可
     */
}
```