|  |  |
| --- | --- |
| **面向对象程序设计 作业报告** | |
| 第1次 | |
| 图片包含 标牌  已生成极高可信度的说明 | |
|  | |
|  | |
|  | |
| **姓名** |  |
| **班级** |  |
| **学号** |  |
| **电话** |  |
| **Email** |
| **日期** | 2022-10-07 |

# 目录

### 题目1 UPC码

1. 题目
2. 数据设计
3. 算法设计
4. 主干代码说明
5. 运行结果展示
6. 总结和收获

### 题目2 数字转英语

1. 题目
2. 数据设计
3. 算法设计
4. 主干代码说明
5. 运行结构展示
6. 总结和收获

### 题目3 长度n的子序列最大乘积

1. 题目
2. 数据设计
3. 算法设计
4. 主干代码说明
5. 运行结构展示
6. 总结和收获

### 题目4 模式化打印图形

1. 题目
2. 数据设计
3. 算法设计
4. 主干代码说明
5. 运行结构展示
6. 总结和收获

### 题目5 直方统计图

1. 题目
2. 数据设计
3. 算法设计
4. 主干代码说明
5. 运行结构展示
6. 总结和收获

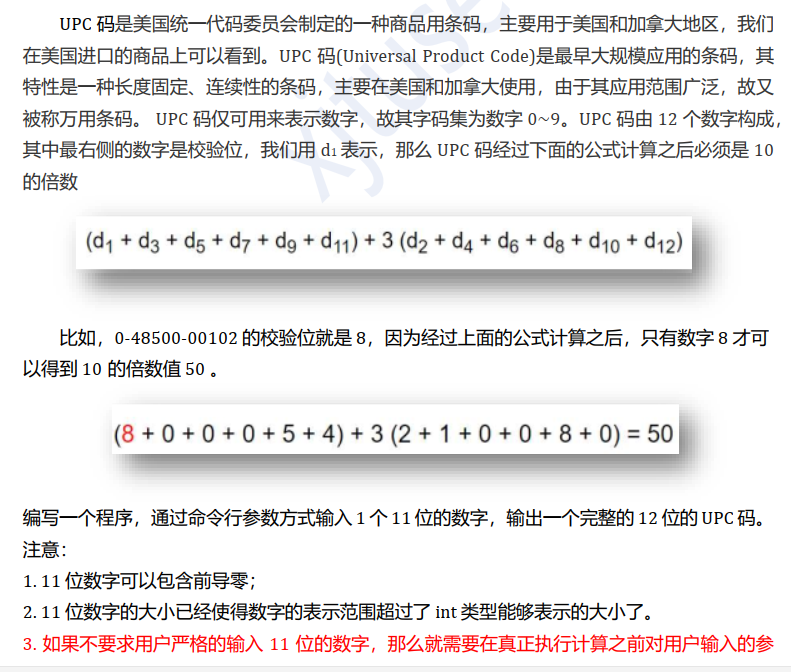
### 题目6 蒙特·卡洛方法模拟

1. 题目
2. 数据设计
3. 算法设计
4. 主干代码说明
5. 运行结构展示
6. 总结和收获

### 附录 每个题的源代码

# 题目1 UPC码

##### 题目





##### 数据设计

本题中，需要存储和处理的数据有：

1. 用户输入的不含校验位的UPC码，由于其大小超过了int类型的表示范围，采取一个long型变量upcIn来存储；
2. 将upcIn复制存储在temp中，以便后续计算使用；
3. 分别计算奇数位、偶数位的和，存储在oddSum和evenSum中；
4. 通过奇数位、偶数位的和计算出校验位，存储在checkBit中；

##### 算法设计

获取输入后，通过循环获取每一位的值，并将偶数位加至偶数位和，将奇数位加至奇数位和；之后再通过一个循环，遍历10的倍数，直到可以解出合适的校验位；最后将校验位在输入后输出即可。

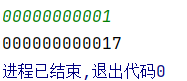
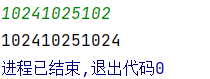
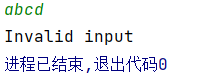
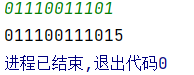
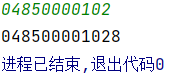
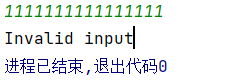
##### 主干代码说明

主干代码如下：

for(int i=2;i<13;i++){  
 if(i%2==1){  
 oddSum+=temp%10;  
 }  
 else{  
 evenSum+=temp%10;  
 }  
 temp/=10;  
}  
*//Calculate the check bit*for(int j=1;checkBit<0;j++){  
 checkBit=10\*j-3\*evenSum-oddSum;  
}  
*//Print the code*System.*out*.printf("%011d%d",upcIn,checkBit);

第一个for循环实现：通过每次取temp除10的余数获得此时的最后一位，判断此时的位是奇数位或偶数位，将其加至对应的和，再将temp除以10以消去已获得的最后一位；第二个for循环实现：分别假设和为10，20，…，直至求出合适的校验位值（即checkBit>=0）为止；输出中的%011d实现强制11位输出，以补上存储至long类型被忽略的先导0。

##### 运行结果展示

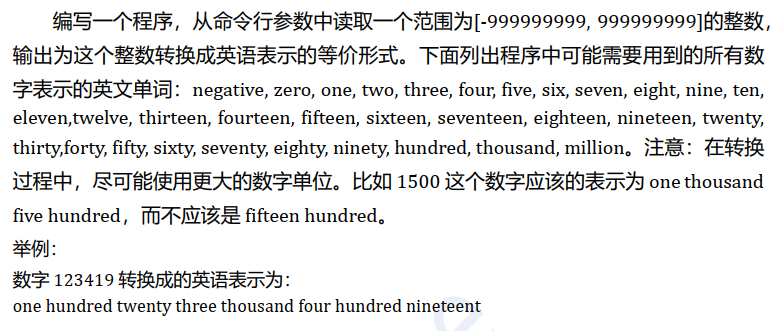


##### 总结和收获

通过本题的实践，增进了我对for循环等的使用能力，并了解了如何通过try...catch进行错误处理，从而处理不合法的输入。

# 题目2 数字转英语

##### 题目



##### 数据设计

本题中，需要存储和处理的数据有：

1. 输入的数字，未超过int范围，使用int类型num表示；
2. 将输入的数字每三位切分，存储在一个int数组num3中；
3. 将数字转化为的英文，由于需要在程序中不断编辑，存储在StringBuilder类型eng中；

##### 算法设计

先判断输入是否为负数或0，分别添加“negative”和“zero”并将负数转为正数；

由于最终的结果呈现（块1）million（块2）thousand（块3）的形式，可将输入的数据每三位分割，分割采取取余或除法的方式；

对于每一个分割出的块，若其百位不为零，添加百位和“hundred”；若其个十位小于20，直接从单词表中添加对应的单词；若其个十位大于20，先添加十位的单词，再添加个位的单词；最后，添加该块对应的单位。

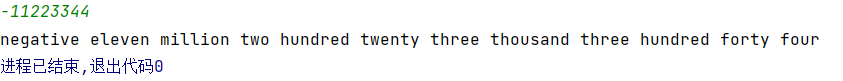
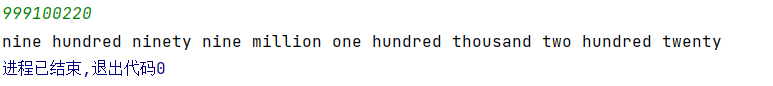
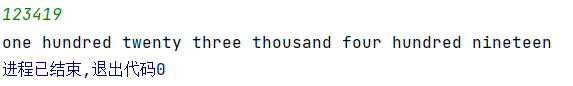
由以上步骤即实现了数字到英语的转换。

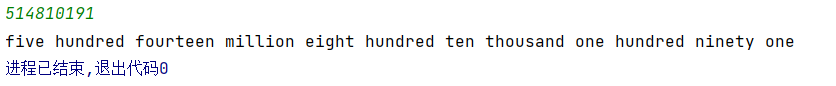
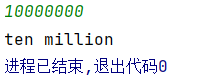
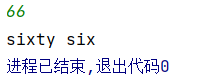
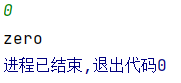
##### 主干代码说明

if(num<0){*//add "negative" to negatives* eng.append("negative ");  
 num=-num;  
}  
if(num==0){*//add "zero" to 0* eng.append("zero");  
}  
*//cut number into pieces*num3[0]=num/1000000;  
num3[1]=(num/1000)%1000;  
num3[2]=num%1000;  
*//deal with the pieces*for(int i=0;i<3;i++){  
 if(num3[i]!=0){  
 if(num3[i]/100!=0){  
 eng.append(*engNum*[num3[i]/100]).append("hundred ");*//add hundreds in each piece* }  
 if(num3[i]%100<20){  
 eng.append(*engNum*[num3[i]%100]);*//<20* }  
 else{  
 eng.append(*engNum*[num3[i]%100-num3[i]%10]).append(*engNum*[num3[i]%10]);*//>20,tens first then ones* }  
 eng.append(*engUnit*[i]);*//add the unit of piece* }  
}

以上即主要实现代码，实现了算法设计中所述的功能。

##### 运行结果展示



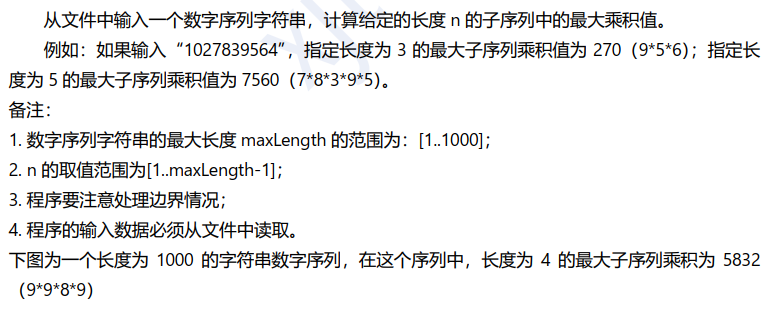


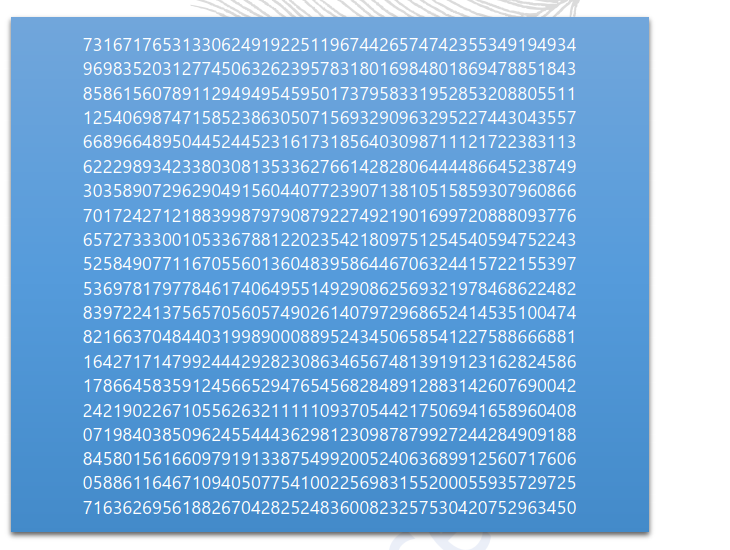
##### 总结和收获

本题目主要让我学到了StringBuilder的用法，并通过对数字的拆分、处理，实现了将数字转换为英文。

# 题目3 长度n的子序列最大乘积

##### 题目





##### 数据设计

本题中，需要存储和处理的数据有：

1. 子序列的长度n
2. 从文件读入的数字序列字符串，采取String类的nums进行存储；
3. 遍历过程中计算出的乘积、最大乘积，分别使用int类型的product、maxProduct进行存储；

##### 算法设计

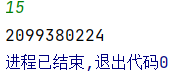
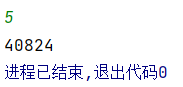
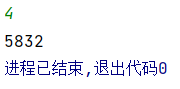
读入子序列的长度n、文件中的数字序列字符串；通过for循环遍历所有子序列，即子序列的最后一位从第n位到最后一位；每一个子序列中，通过for循环求得子序列每一位的乘积，若求得的乘积比当前最大乘积大，则更新最大乘积。

##### 主干代码说明

for(int i=n-1;i<nums.length();i++){  
 product=1;  
 for(int j=0;j<n;j++){*//Calculate each product* product\*=Integer.*parseInt*(nums.substring(i-j,i-j+1));  
 }  
 if(product>maxProduct){*//If the result is bigger than the max, replace the max* maxProduct=product;  
 }  
}

以上主干代码实现了算法设计中所述的两个for循环。其中，nums.substring(i-j,i-j+1)取得字符串中第i-j位小字符串，Integer.*parseInt*实现字符串转换为数字，从而求得乘积。

##### 运行结果展示



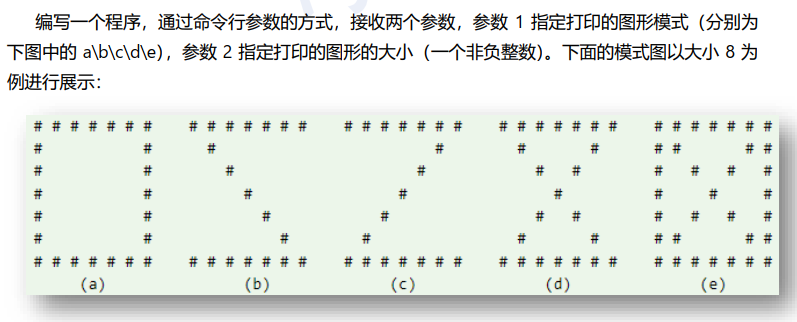
注：文件中的数字序列字符串与题目所给示例相同。

##### 总结和收获

通过本题，练习了从文件读取数据、提取字符串的一部分、将字符串转换为数字等，通过循环嵌套实现计算出子序列的最大乘积。

# 题目4 模式化打印图形

##### 题目



##### 数据设计

本题目主要涉及的数据有：

1. 输入的代表类型的字母，使用String type存储；
2. 输入的大小数字，使用int size存储；
3. 根据type确定的在不同位置是否输出的存储，使用char[] charByType存储，有# ， # ， #， ##，###五种组合；

##### 算法设计

根据输入的类型确定在两侧、左对角线、右对角线处是否输出，具体在charByType变量中分别体现为’#’和空格；输出环节第一行、最后一行利用for循环直接输出；中间部分利用row变量控制行数遍历，在每一行中使用多个for循环和min、max函数控制空格的输出。

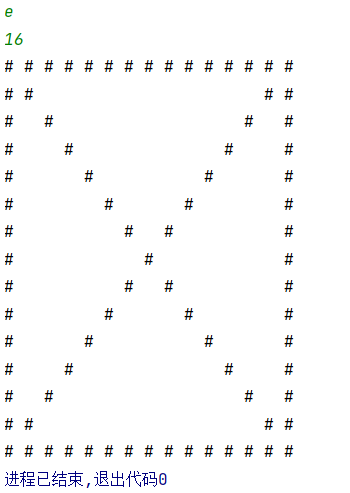
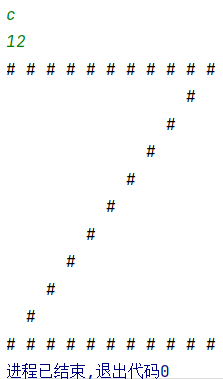
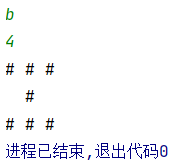
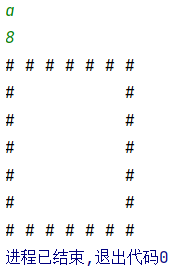
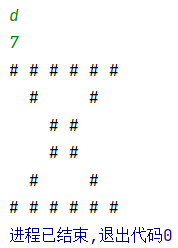
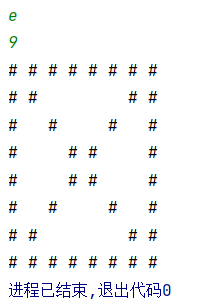
##### 主干代码说明

for (int row = 2; row < size - 1; row++) {  
 System.*out*.print(charByType[0]);  
 for (int i = 1; i < Math.*min*(2\*row-2, 2\*(size-row-1)); i++) {  
 System.*out*.print(" ");  
 }  
 if(row - 1 != size - row - 1){*//normal* if(2\*row-2<2\*(size-row-1)){  
 System.*out*.print(charByType[1]);  
 }  
 else{  
 System.*out*.print(charByType[2]);  
 }  
 for (int i = Math.*min*(2\*row-2, 2\*(size-row-1)) + 1; i < Math.*max*(2\*row-2, 2\*(size-row-1)); i++) {  
 System.*out*.print(" ");  
 }  
 if(2\*row-2<2\*(size-row-1)){  
 System.*out*.print(charByType[2]);  
 }  
 else{  
 System.*out*.print(charByType[1]);  
 }  
 }  
 else{*//position of charByType[1]=position of charByType[2]* if(charByType[2]=='#'||charByType[1]=='#'){  
 System.*out*.print('#');  
 }  
 else{  
 System.*out*.print(' ');  
 }  
 }  
 for (int i = Math.*max*(2\*row-2, 2\*(size-row-1)) + 1; i < 2\*size - 4; i++) {  
 System.*out*.print(" ");  
 }  
 System.*out*.print(charByType[0]);  
 System.*out*.println();  
}

以上为输出图形中间行的代码。

1. 最前和最后的System.*out*.print(charByType[0]);控制行两侧的输出；
2. for (int i = 1; i < Math.*min*(2\*row-2, 2\*(size-row-1)); i++)控制左侧到左最近对角线的空格；
3. if(row - 1 != size - row - 1)…else…判断两对角线是否相交，若相交执行特殊输出方式；
4. if(2\*row-2<2\*(size-row-1)){System.*out*.print(charByType[1]);}else{System.*out*.print(charByType[2]);判断左最近对角线为左对角线或右对角线，进行对应的输出；
5. for (int i = Math.*min*(2\*row-2, 2\*(size-row-1)) + 1; i < Math.*max*(2\*row-2, 2\*(size-row-1)); i++)控制两对角线之间的空格；
6. for (int i = Math.*max*(2\*row-2, 2\*(size-row-1)) + 1; i < 2\*size - 4; i++)控制右最近对角线至右端的空格。

##### 运行结果展示



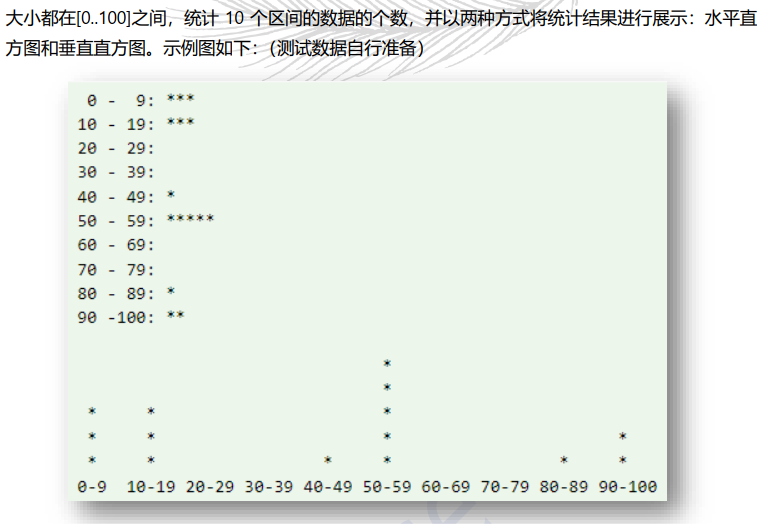
##### 总结和收获

本题是所有题目中耗时最长的一道，困难点主要在于如何控制输出不同格式的输出和在每一行的输出中如何进行空格的控制、字符的控制。最终采取了char数组的方式来分别控制两端、对角线的输出，但对于行中空格、字符输出的控制未能想到较好的解决方法，因此最终呈现的程序较为复杂。

# 题目5 直方统计图

##### 题目





##### 数据设计

本题主要涉及的数据有：

1. 组名称，即“0-9”“10-19”，存储在String[] nameOfGroup中方便输出时调用；
2. 组个数，即各组包含的数据个数，存储在int[] numOfGroup中；
3. 从文件读入的数据，采取直接处理不存储的方式；

##### 算法设计

1. 首先从文件读入数据，将数据分组个数统计至int[] numOfGroup；
2. 之后输出行直方图，利用循环在每行分别输出组名，再利用一个循环输出对应个数的\*；
3. 再次输出列直方图，需要获取组个数的最大值，利用循环从最大值遍历至1即为输出数据的行。在每一行中，确定所对应列在该行是否有输出；
4. 最后还需实现上述最大值功能，即获取一数组内的最大值。

##### 主干代码说明

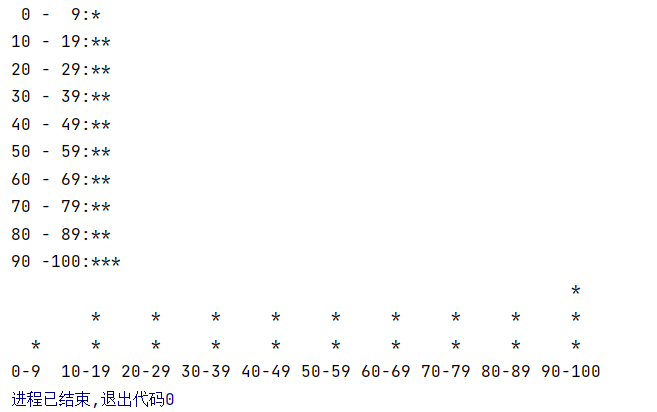
while(in.hasNext()){  
 numOfGroup[in.nextInt()/10]++;*//Scan the data and divide into groups*}  
numOfGroup[9]+=numOfGroup[10];*//When data is 100,it adds to numOfGroup[10]*

以上是文件输入部分代码。in.hasNext()保证循环在文件中还有数据时继续，每次循环中将对应数据除10，将对应的numOfGroup 加一即可。最后，需要将100对应的numOfGroup[10]加至numOfGroup[9]。

*//print row histogram* for(int i=0;i<10;i++){  
 System.*out*.printf("%s:",nameOfGroup[i]);*//Print name* for(int j=0;j<numOfGroup[i];j++) {  
 System.*out*.print("\*");*//Print "\*"* }  
 System.*out*.println();  
 }  
 *//print column histogram* for(int j=*getMax*(numOfGroup);j>0;j--){  
 System.*out*.print(" ");  
 for(int k=0;k<10;k++){  
 if(numOfGroup[k]>=j){*//Confirm the printing for each group* System.*out*.print("\*");  
 }  
 else{  
 System.*out*.print(" ");  
 }  
 System.*out*.print(" ");  
 }  
 System.*out*.println();  
 }  
 System.*out*.print("0-9 10-19 20-29 30-39 40-49 50-59 60-69 70-79 80-89 90-100");  
}

以上是输出直方图的代码。在输出列直方图时，每行先输出2个空格，之后进入循环开始判断，若当前对应列的值小于等于当前行高，则输出\*，否则输出空格；每次判断结束后输出5个空格作为分隔。

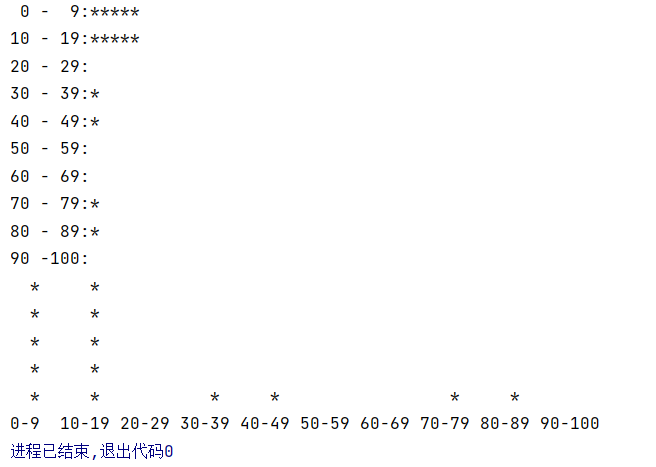
##### 运行结果展示



数据：100 10 20 30 40 50

60 70 80 90 99 88 77 66

55 43 32 21 10 0



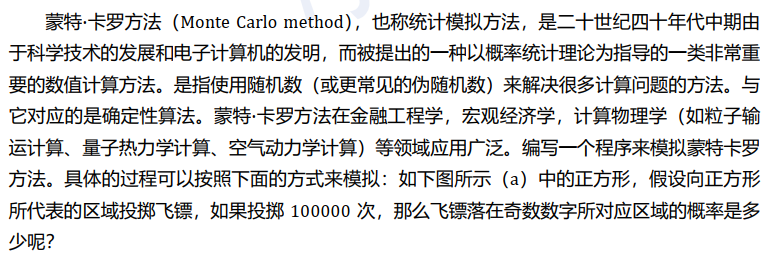
数据：11 4 5 10 19 19 81 0 33 44 70 10 0 5

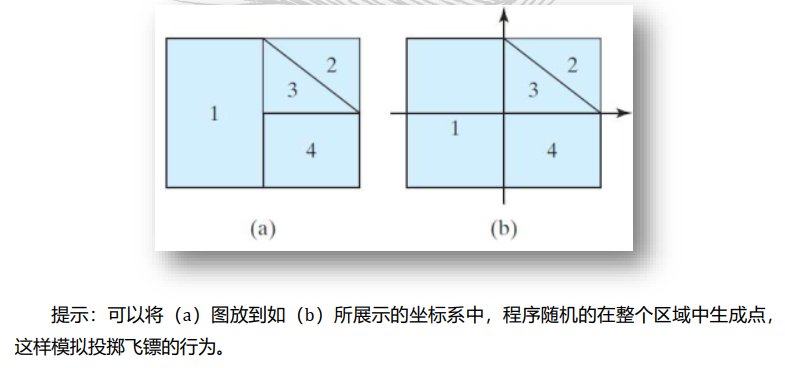
##### 总结和收获

本题主要练习了数据分类处理、利用循环和条件判断控制输出等，成功实现了绘制简单的直方图。

# 题目6 蒙特·卡洛方法模拟

##### 题目





##### 数据设计

本题涉及到的数据有：

1. 每次随机生成点的两个坐标，采取double类型变量存储；
2. 点落在奇数区域的次数，，采取int类型变量存储；

##### 算法设计

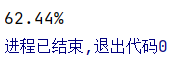
利用for循环，随机生成坐标100000次，每次循环中判断点是否落在奇数区域中，若在则将区域内次数加一；最后以百分比形式输出区域内次数与总次数之比即可。

##### 主干代码说明

for(int i=0;i<times;i++){  
 x=r.nextDouble()\*2-1;  
 y=r.nextDouble()\*2-1;*//Generate random x,y  
 //Area 1* if(x<0){  
 timesInArea++;  
 }  
 *//Area 3* else if(x>0 && y>0 && x+y<1){  
 timesInArea++;  
 }  
 }  
 System.*out*.printf("%.2f%%",(float)timesInArea/(float)times\*100);

主干代码如上所示，r.nextDouble()随机生成一个0到1之间的浮点数，通过将之乘2减1获得-1到1之间的浮点数；由图示，若x坐标小于0则点落在1区域，若xy坐标均大于0且x+y小于1则点落在3区域；循环结束后输出区域内次数与总次数的比。

##### 运行结果展示



##### 总结和收获

根据概率统计知识可知，点落在奇数区域内的概率为62.50%，程序运行结果都在该值附近，与之相符。

本题主要练习了通过random取随机数、if判断点是否在区域内等。

# 附录 每个题的源代码

##### 题目1

import java.util.Scanner;  
public class UPC{  
 public static void main(String[] args){  
 long upcIn=0,temp=0;  
 int evenSum=0,oddSum=0;  
 int checkBit=-1;  
 Scanner in = new Scanner(System.*in*);  
 try{*//"try...catch" to deal with invalid input* upcIn=in.nextLong();*//Scan the input* if(upcIn>99999999999l){  
 throw new Exception("Too big input");  
 }  
 *//Sum the even bits and the odd bits* temp=upcIn;  
 for(int i=2;i<13;i++){  
 if(i%2==1){  
 oddSum+=temp%10;  
 }  
 else{  
 evenSum+=temp%10;  
 }  
 temp/=10;  
 }  
 *//Calculate the check bit* for(int j=1;checkBit<0;j++){  
 checkBit=10\*j-3\*evenSum-oddSum;  
 }  
 *//Print the code* System.*out*.printf("%011d%d",upcIn,checkBit);  
 }catch(Exception e){  
 System.*out*.print("Invalid input");  
 }  
 }  
}

##### 题目2

import java.util.Scanner;  
public class NumToEng {  
 public static final String[] *engNum* = {  
 "", "one ", "two ", "three ", "four ", "five ", "six ", "seven ", "eight ", "nine ",  
 "ten ", "eleven ", "twelve ", "thirteen ", "fourteen ", "fifteen ", "sixteen ", "seventeen ", "eighteen ", "nineteen ",  
 "twenty ", "", "", "", "", "", "", "", "", "",  
 "thirty ", "", "", "", "", "", "", "", "", "",  
 "forty ", "", "", "", "", "", "", "", "", "",  
 "fifty ", "", "", "", "", "", "", "", "", "",  
 "sixty ", "", "", "", "", "", "", "", "", "",  
 "seventy ", "", "", "", "", "", "", "", "", "",  
 "eighty ", "", "", "", "", "", "", "", "", "",  
 "ninety " };*//words which could be used* public static final String[] *engUnit* = {"million ","thousand ",""};  
 public static void main(String[] args){  
 int num;  
 int[] num3=new int[3];*//num3 is used to storage numbers cut in pieces* StringBuilder eng=new StringBuilder();*//eng is used to storage eng results* Scanner in = new Scanner(System.*in*);  
 num=in.nextInt();*//input* if(num<0){*//add "negative" to negatives* eng.append("negative ");  
 num=-num;  
 }  
 if(num==0){*//add "zero" to 0* eng.append("zero");  
 }  
 *//cut number into pieces* num3[0]=num/1000000;  
 num3[1]=(num/1000)%1000;  
 num3[2]=num%1000;  
 *//deal with the pieces* for(int i=0;i<3;i++){  
 if(num3[i]!=0){  
 if(num3[i]/100!=0){  
 eng.append(*engNum*[num3[i]/100]).append("hundred ");*//add hundreds in each piece* }  
 if(num3[i]%100<20){  
 eng.append(*engNum*[num3[i]%100]);*//<20* }  
 else{  
 eng.append(*engNum*[num3[i]%100-num3[i]%10]).append(*engNum*[num3[i]%10]);*//>20,tens first then ones* }  
 eng.append(*engUnit*[i]);*//add the unit of piece* }  
 }  
 System.*out*.print(eng);*//output* }  
}

##### 题目3

import java.io.File;*//needed for file input*import java.io.FileNotFoundException;  
import java.util.Scanner;  
public class SubMaxProduct {  
 public static void main(String[] args) throws FileNotFoundException {  
 String nums;  
 int n;*//n is the length of sub string* int maxProduct= 0;  
 int product;  
 Scanner in=new Scanner(new File("in.txt"));  
 Scanner sysin=new Scanner(System.*in*);  
 nums=in.next();*//file input* n=sysin.nextInt();*//n input* for(int i=n-1;i<nums.length();i++){  
 product=1;  
 for(int j=0;j<n;j++){*//Calculate each product* product\*=Integer.*parseInt*(nums.substring(i-j,i-j+1));  
 }  
 if(product>maxProduct){*//If the result is bigger than the max, replace the max* maxProduct=product;  
 }  
 }  
 System.*out*.print(maxProduct);*//output* }  
}

##### 题目4

import java.util.Scanner;  
public class PrintPattern {  
 public static void main(String[] args) {  
 String type;  
 int size;  
 Scanner in = new Scanner(System.*in*);  
 type = in.next();*//scan type* size = in.nextInt();*//scan size* char[] charByType = switch (type) {*//decide char output in positions* case "a" -> new char[]{'#', ' ', ' ' };  
 case "b" -> new char[]{' ', '#', ' ' };  
 case "c" -> new char[]{' ', ' ', '#' };  
 case "d" -> new char[]{' ', '#', '#' };  
 case "e" -> new char[]{'#', '#', '#' };  
 default->new char[]{' ', ' ', ' ' };  
 };  
 *//First line:* for (int i = 1; i < size; i++) {  
 System.*out*.print("# ");  
 }  
 System.*out*.println();  
 *//Middle line:* for (int row = 2; row < size - 1; row++) {  
 System.*out*.print(charByType[0]);  
 for (int i = 1; i < Math.*min*(2\*row-2, 2\*(size-row-1)); i++) {  
 System.*out*.print(" ");  
 }  
 if(row - 1 != size - row - 1){*//normal* if(2\*row-2<2\*(size-row-1)){  
 System.*out*.print(charByType[1]);  
 }  
 else{  
 System.*out*.print(charByType[2]);  
 }  
 for (int i = Math.*min*(2\*row-2, 2\*(size-row-1)) + 1; i < Math.*max*(2\*row-2, 2\*(size-row-1)); i++) {  
 System.*out*.print(" ");  
 }  
 if(2\*row-2<2\*(size-row-1)){  
 System.*out*.print(charByType[2]);  
 }  
 else{  
 System.*out*.print(charByType[1]);  
 }  
 }  
 else{*//position of charByType[1]=position of charByType[2]* if(charByType[2]=='#'||charByType[1]=='#'){  
 System.*out*.print('#');  
 }  
 else{  
 System.*out*.print(' ');  
 }  
 }  
 for (int i = Math.*max*(2\*row-2, 2\*(size-row-1)) + 1; i < 2\*size - 4; i++) {  
 System.*out*.print(" ");  
 }  
 System.*out*.print(charByType[0]);  
 System.*out*.println();  
 }  
 *//Last line:* for (int i = 1; i < size; i++) {  
 System.*out*.print("# ");  
 }  
 }  
}

##### 题目5

import java.io.File;*//needed for file input*import java.io.FileNotFoundException;  
import java.util.Scanner;  
public class Histogram {  
 public static void main(String[] args) throws FileNotFoundException {  
 String[] nameOfGroup={" 0 - 9","10 - 19","20 - 29","30 - 39","40 - 49","50 - 59","60 - 69","70 - 79","80 - 89","90 -100"};  
 int[] numOfGroup=new int[11];*//Storage how many elements each group has* Scanner in=new Scanner(new File("HistogramIn.txt"));  
 while(in.hasNext()){  
 numOfGroup[in.nextInt()/10]++;*//Scan the data and divide into groups* }  
 numOfGroup[9]+=numOfGroup[10];*//When data is 100,it adds to numOfGroup[10]  
 //print row histogram* for(int i=0;i<10;i++){  
 System.*out*.printf("%s:",nameOfGroup[i]);*//Print name* for(int j=0;j<numOfGroup[i];j++) {  
 System.*out*.print("\*");*//Print "\*"* }  
 System.*out*.println();  
 }  
 *//print column histogram* for(int j=*getMax*(numOfGroup);j>0;j--){  
 System.*out*.print(" ");  
 for(int k=0;k<10;k++){  
 if(numOfGroup[k]>=j){*//Confirm the printing for each group* System.*out*.print("\*");  
 }  
 else{  
 System.*out*.print(" ");  
 }  
 System.*out*.print(" ");  
 }  
 System.*out*.println();  
 }  
 System.*out*.print("0-9 10-19 20-29 30-39 40-49 50-59 60-69 70-79 80-89 90-100");  
 }  
 public static int getMax(int[] array){  
 int max=0;  
 for(int i=0;i<10;i++){  
 if(array[i]>max){  
 max=array[i];  
 }  
 }  
 return max;  
 }  
}

##### 题目6

import java.util.Random;  
public class MonteCarloSimulation {  
 public static void main(String[] args){  
 Random r=new Random();  
 double x,y;  
 int timesInArea=0,times=100000;  
 for(int i=0;i<times;i++){  
 x=r.nextDouble()\*2-1;  
 y=r.nextDouble()\*2-1;*//Generate random x,y  
 //Area 1* if(x<0){  
 timesInArea++;  
 }  
 *//Area 3* else if(x>0 && y>0 && x+y<1){  
 timesInArea++;  
 }  
 }  
 System.*out*.printf("%.2f%%",(float)timesInArea/(float)times\*100);  
 }  
}