**《算法设计与分析》实验报告**

# 实验六 概率算法

|  |  |  |  |
| --- | --- | --- | --- |
| **报告书** | | | |
| 姓名 | 吴宇敖 | 指导教师 |  |
| 学号 | 176001752 | 日 期 | 2020.6.4 |
| 班级 | 17计算机软件一班 |  |  |
| **实验内容** | | | |
| **[本次实验无OJ测试]**   1. 使用随机方法求解圆周率。 2. 使用舍伍德算法求n个数中的第k大的数。 3. 试用拉斯维加斯方法求解8皇后问题。 4. 举例说明蒙特卡洛方法的应用。 | | | |
| **实验目的** | | | |
| 1）熟悉和掌握随机化方法  2）熟悉和掌握舍伍德算法、拉斯维加斯算法、蒙特卡罗算法以及它们的特点。 | | | |
| **实验过程和步骤** | | | |
| **以下仅供参考，请在报告中删除**  **【实验题目】**  使用随机方法求解圆周率。  **【程序代码及注释】**  #include<iostream>  #include<random>  using namespace std;  int main()  {  double n;  int r;  cout<<"请输入测试园的半径:(小于32767)"<<endl;  cin>>r;  cout<<"请输入测试点的数目:"<<endl;  cin>>n;  int x,y;  double sum=0;//用来记录在圆里面的点  for(int i=0;i<n;i++)  {  x=rand()%r;//随机x的值  y=rand()%r;//随机y的值  if(((x\*x)+(y\*y))<=(r\*r))//判断点在不在圆里面  sum++;  }  cout<<(4\*sum/n)<<endl;//输出π  system("pause");  return 0;  }  **【实验结果及遇到的问题】**    遇到的问题：一开始我为例防止随机数据的相同，加了 srand( (unsigned)time( NULL ) );这样就会导致最后算出来的结果不是4就是0  **【算法分析】**  我是按照随机点在一个正方形和它的内切扇形中随机落到扇形区域的概率来进行π值的计算，使用了sum来记录落到扇形圆内的概率，n表示一共有几个点，最后π值就为4\*sum/n  **【实验题目】**  使用舍伍德算法求n个数中的第k大的数。  **【程序代码及注释】**  #include<iostream>  #include<random>  using namespace std;  int num[1000];//存储数据  int quick(int left,int right ,int m)//快排  {  if(left==right)  return left;  int k=left+rand()%(right-left);//基准数定义为随机数  int i=left;  int j=right;  int key=num[k];  while (1)//i从前向后，j从后向前  {  while (num[i] <= key)//寻找比key大的数  {  if (i == right){  break;  }  i++;  }  while (num[j] >= key)//寻找比key小的数  {  if (j == left){  break;  }  j--;  }  if (i >= j)  break;  swap(num[i],num[j]);//交换  }  if(i<=k)  {  swap(num[i],num[k]);//将基准数放置到合适位置  if(i+1==m)//找到第m大的数，返回  return i;  else if(i+1>m)  quick(left,i-1,m);  else quick(i+1,right,m);  }  else {  swap(num[j],num[k]);//将基准数放置到合适位置  if(j+1==m)//找到第m大的数，返回  return j;  else if(j+1>m)  quick(left,j-1,m);  else quick(j+1,right,m);  }  }  int main()  {  int n,m;  cin>>n>>m;  for(int i=0;i<n;i++)  cin>>num[i];  cout<<num[quick(0,n-1,m)]<<endl;  system("pause");  return 0;  }  **【实验结果及遇到的问题】**      **【算法分析】**  这个实验就是通过快速排序来寻找第k大的数，由于使用舍伍德算法，所以快速排序的基准数应该是随机产生的。我通过调用随机函数来随机出一个在开始和终止节点之间的一个基准数。之后每次快速排序都判断第m大的数在随机出的基准数的哪一边，若就是基准数则返回值，不然的话就对第m大的数在的那一边进行递归。  **【实验题目】**  试用拉斯维加斯方法求解8皇后问题。  **【程序代码及注释】**  #include<iostream>  #include<time.h>  using namespace std;  int num[20]={0};  bool chick(int n)//来进行是否满足每个皇后都不相吃的判断  {  for(int i=0;i<n;i++)  for(int j=i+1;j<n;j++)  {  if(num[i]==num[j]||((num[i]-num[j])==(j-i))||((num[j]-num[i])==(j-i)))  return false;  }  return true;  }  void find(int n)//不断进行循环直到找到一种可能  {  while(1)  {  //srand(time(NULL));  for(int i=0;i<n;i++)  {  num[i]=rand()%n;  }  if(chick(n))  break;  }  }  int main()  {  cout<<"请输入n，n为nXn的棋盘(n小于20)"<<endl;  int n;  cin>>n;  find(n);  for(int i=0;i<n;i++)//进行输出  {  for(int j=0;j<n;j++)  {  if(num[i]==j)  cout<<setw(5)<<num[j]+1;  else cout<<setw(5)<<"0";  }  cout<<endl;  }  system("pause");  return 0;  }  **【实验结果及遇到的问题】**      **【算法分析】**  这次实验是使用拉斯维加斯方法求解8皇后问题，我是通过全随机的方法来对每个皇后的位置进行随机，然后再进行判断，是否满足题目需求，满足要求就退出循环，如果不满足就继续随机，知道满足要求为止。  **【实验题目】**  举例说明蒙特卡洛方法的应用。  **【程序代码及注释】**  #include<iostream>  #include<time.h>  #include<math.h>  using namespace std;  long pow(int x,int y)//计算x的y次  {  long res=1;  for(int i=0;i<y;i++)  res=res\*x;  return res;  }  bool test(int x)//使用费尔马小定理，进行初步判断  {  srand(time(NULL));  int a=rand()%(x-1)+1;  long b=pow(a,x-1);  if(b%x==1)  return true;  else return false;  }  bool againtest(int x,int n)//由于费尔马小定理只是一个必要条件，所以需要使用二次探测定理来进行二次判断  {  int num=0;  for(int i=0;i<n;i++)  if(test(x))  num++;  if(num>=n\*0.9)  for(int i=0;i<n;i++)  {  srand(time(NULL));  int a=rand()%(x-1)+1;  if(((a\*a)-1)%x==0&&(a!=1&&a!=x-1))  return false;  else continue;  }  else return false;  return true;  }  int main()  {  int x;  cout<<"输入一个数:"<<endl;  cin>>x;  if(againtest(x,1000))  cout<<"是素数"<<endl;  else cout<<"不是素数"<<endl;  system("pause");  return 0;  }  **【实验结果及遇到的问题】**        遇到问题：对于素数检测的哪几种方法我不怎么清楚，通过网上资料的查询我最后才明白了素数检测需要进行的步骤  **【算法分析】**  先使用费尔马小定理，进行初步判断。由于费尔马小定理只是一个必要条件，并且它返回的值并不是完全正确，所以我先对于费尔马小定理来进行1000次判断，如果判断的结果为true的值的数量小于90%，那么直接判断为非素数，如果超过90%，那么就进入到二次探测定理来进行二次判断，通过二次探测定理可以知道（x\*x）%n=1的解为1或者是n-1，通过这个条件来进行多次循环判断是否为素数。 | | | |