3.Data Structure Basic Concepts

1 Basic Concepts

https://www.tutorialspoint.com/data_structures_algorithms/data_structures_basics.htm

This chapter explains the basic terms related to data structure.

Data Definition

Data Definition defines a particular data with the following characteristics.

- **Atomic** − Definition should define a single concept.
- **Traceable** − Definition should be able to be mapped to some data element.
- **Accurate** − Definition should be unambiguous.
- **Clear and Concise** − Definition should be understandable.

Data Object

Data Object represents an object having a data.

Data Type

Data type is a way to classify various types of data such as integer, string, etc. which determines the values that can be used with the corresponding type of data, the type of operations that can be performed on the corresponding type of data. There are two data types −

- Built-in Data Type
- Derived Data Type

Built-in Data Type

Those data types for which a language has built-in support are known as Built-in Data types. For example, most of the languages provide the following built-in data types.

- Integers
- Boolean (true, false)
- Floating (Decimal numbers)
- Character and Strings

Derived Data Type

Those data types which are implementation independent as they can be implemented in one or the other way are known as derived data types. These data types are normally built by the combination of primary or built-in data types and associated operations on them. For example −

- List
- Array
- Stack
- Queue

Basic Operations

The data in the data structures are processed by certain operations. The particular data structure chosen largely depends on the frequency of the operation that needs to be performed on the data structure.

- Traversing
- Searching
- Insertion
- Deletion
- Sorting
- Merging

2.Array

（1）埃拉托色尼算法

功能：如果自然数为素数，a[i]=1,否则a[i]=0。

- 把数组中所有元素设为1，以表明没有任何数被证明为非素数
- 把数组中所有对应索引处已证明是非素数（已知素数的倍数）的元素设为0.如果所有更小的倍数都已经设为0，a[i]仍然为1，则可知它是素数。

```
#include<stdio.h>
#define N 10000
int main ()
{
    int i,j,a[N];
    for(i=2;i<N;i++)
        a[i]=1;//把数组中所有元素设为1，以表明没有任何数被证明
为非素数
    for(i=2;i<N;i++)
        if(a[i])
          for(j=i;j*i<N;j++)a[j*i]=0;//所有小于10000，某数
的倍数，都不是素数。
    for(i=2;i<N;i++)
        if(a[i])printf("%8d",i);
    return 0;
}
```

（2）数组的动态存储分配——抛硬币模拟试验（Bernoulli trial)

上面（1）中的最大数N为已知，而若要改变N，只能修改程序重新编译。另一种方法是：程序从命令行中获取期望的最大数，再通过stdilb.c的malloc，利用该值为数组动态分配内存.

下面举一个实例：伯努利实验：如果我们抛一枚硬币N次，那么看到k次正面的概率是

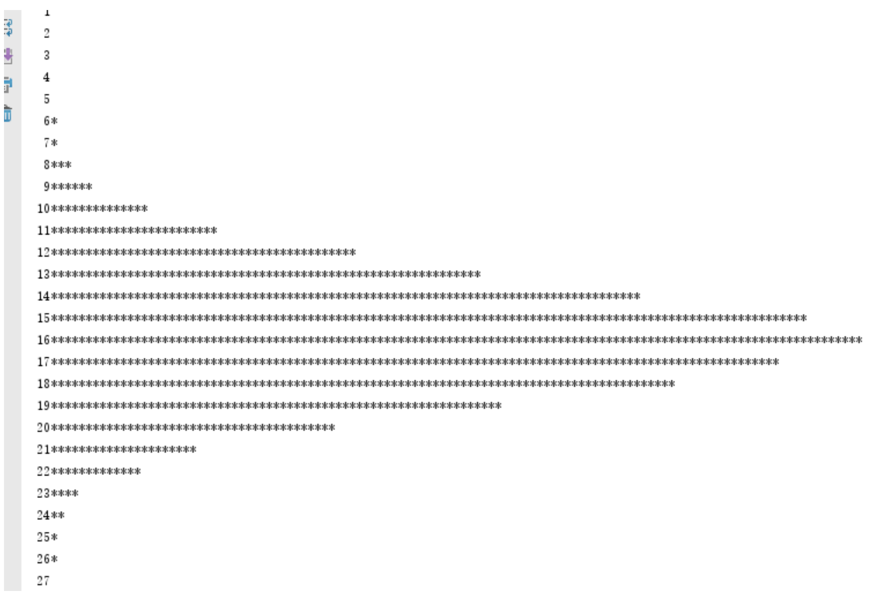
$$\heartsuit\heartsuit\heartsuit\binom{N}{k}\frac{1}{2^N}\approx\frac{e^{-(k-N/2)^2/N}}{\sqrt{\pi N/2}}\heartsuit\heartsuit\heartsuit$$

(L ATEX2ε)http://mohu.org/info/symbols/symbols.htm

```c
#include<stdio.h>
#include<stdlib.h>
int heads()
{
    return rand() <RAND_MAX/2;//小于返回1，不小于返回0，概率
1/2
}
int main()
{
    int M,N;
    int i ,j,cnt=0;
    scanf("%d",&M);//输入抛硬币的次数N和实验的次数M
    scanf("%d",&N);
    int *f=malloc((N+1)*sizeof(int));//动态申请内存储存存储
数据
    for(j=0;j<=N;j++)
        f[j]=0;
    for(i=0;i<M;i++,f[cnt]++)
        for(cnt=0,j=0;j<N;j++) //类似于桶排序
           if(heads()) cnt++;
    for(j=0;j<=N;j++)
    {
        printf("%2d",j);
        for(i=0;i<f[j];i+=10) printf("*");//以10为单位，输
出出现j次的次数
        printf("\n");
    }
    return 0;

}
```

当M=8000，N=32的时候，结果非常满足动态分布



```
 1
 2
 3
 4
 5
 6*
 7*
 8***
 9******
10**************
11************************
12********************************************
13**************************************************************
14************************************************************************************
15************************************************************************************************************
16********************************************************************************************************************
17********************************************************************************************************
18*****************************************************************************************
19****************************************************************
20*****************************************
21*********************
22*************
23****
24**
25*
26*
27
```

(3)最近点对的计算

对N个随机产生的单位正方形中的点，统计可以被长度小于d的直线连结的点的对数。

```
#include<stdio.h>
#include<stdlib.h>
#include<math.h>
typedef struct
{
    float x;
    float y;
}Point;//定义一个结构体存放点的横纵坐标
float randFloat()
{
    return 1.0*rand()/RAND_MAX;随机产生一个[0,1]的浮点数
}
float distance( Point a, Point b)
{
    float dx = a.x-b.x,dy=a.y-b.y;
    return sqrt(dx*dx+dy*dy);
}//计算两点之间的距离
int main()
{
    float d;
    int N;
    int i ,j,cnt=0;
    scanf("%f",&d);//读入距离
    scanf("%d",&N);//点的个数
    Point *a=malloc(N*sizeof(*a));//结构体数组
    for(i=0;i<N;i++)
    {
        a[i].x=randFloat();
        a[i].y=randFloat();
    }
    for(i=0;i<N;i++)
    {
        for(j=i+1;j<N;j++)
        {
            if(distance(a[i],a[j])<d) cnt++;
        }
    }
    printf("%d edges shorter than %f\n",cnt,d);
    return 0;

}
```

虽然枚举很暴力，但是我们这里主要学了创建任意复杂类型的方法：可以是结构体数组，可以是数组组成的数组，或者包含数组的结构体。之后我们会仔细考虑各种数组的用法，但是下一章，还是不得不来链表了啊。。。表示我看了好多本关于数据结构的书都只能看到链表。。。。。但是这是除数组外另一种组织对象集合的主要方法。