线性结构

计算机程序设计 -- 数据结构

重庆南开信竞基础课程 Helang

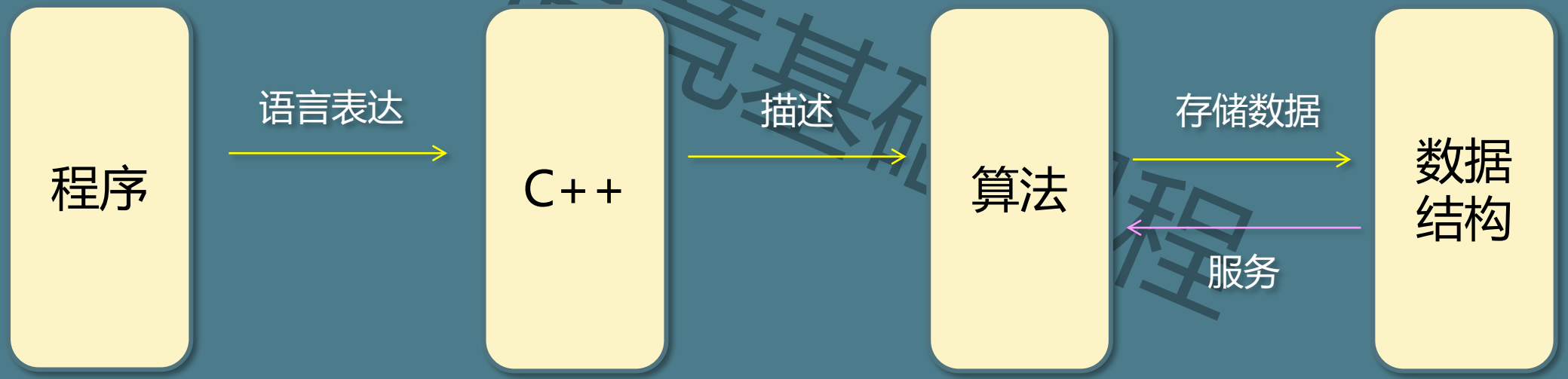
Which Challenges Are the Most Popular?

Percentage of HackerRank Tests Taken By Type

Rank	Domain	Percent of Tests
1	Algorithms	39.5%
2	Java	9.3%
3	Data Structures	9.1%
4	C++	6.6%
5	Tutorials	6.5%
6	Mathematics	6.1%
7	Python	5.3%
8	SQL	4.6%
9	Shell	3.1%
10	Artificial Intelligence	2.9%
11	Functional Programming	2.5%
12	Databases	1.5%
13	Ruby	1.0%
14	Distributed Systems	1.0%
15	Security	0.9%
	Total	100.0%

什么是数据结构?

数据结构(Data Structure),用于描述计算机中数据的存储、组织形式。合理的数据结构可以给程序带来更高的存储和运行效率。



什么是数据结构?

数据结构(Data Structure),用于描述计算机中数据的存储、组织形式。合理的数据结构可以给程序带来更高的存储和运行效率。

常用的数据结构有哪些?

1.线性结构　栈、队列、链表

2.树型结构

3.图型结构

1.队列

队首（front）

队列

队尾（back）

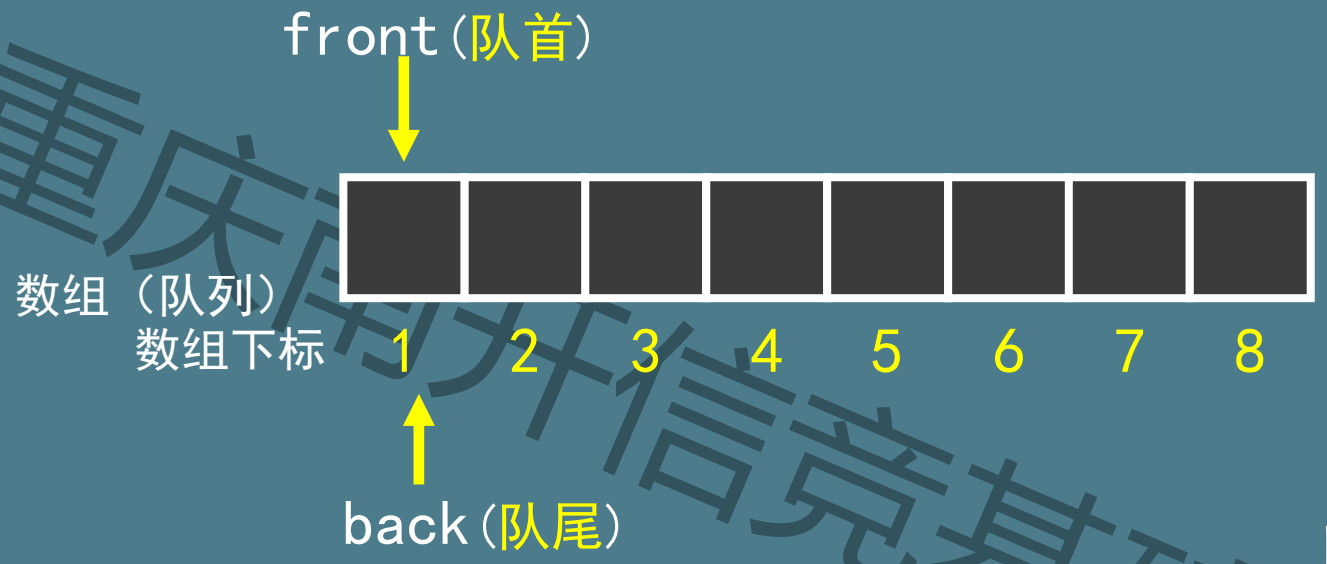
队列（queue）是另一种特殊的线性表，它的特点是删除操作在一头进行，插入操作在另一头进行。

允许删除的一端称为队首（front），允许插入的一端称为队尾(back)。

不含任何元素的队称为空队。

队列的修改是按先进先出(First In First Out)的原则进行

用数组模拟队列的“先进先出”



当front==back
时表示队列为空

1. 插入数据：C
2. 插入数据：F
3. 插入数据：X
4. 删除数据
4. 删除数据
5. 插入数据：M

队列的代码实现

```
#define maxn 101
char Queue[maxn];
int front,back;
```

```
//入队
void insert(char x)
{
  if(back>=maxn)cout<<"full";
  else
  {
        Queue[back]=x;
        back++;
   }
}
```

```
//出队
void del()
{
if(back==front)cout<<"empty";
  else
  {
        cout<<Queue[front];
        front++;
    }
}
```

queue

queue 队列

queue（队列），插入只可以在尾部进行，删除、检索和修改只允许从头部进行。
按照先进先出的原则。

queue<数据类型> 队列名 - 声明队列
比如 queue<int>q; 表示声明了一个叫q的只能存整数的队列

常用函数:
push(e) - 将元素e加入队列尾部,例: q.push(5);
pop() - 将队首元素删除,例: q.pop();
front() - 返回队首元素的值,例: cout<<q.front();
back() - 返回队尾元素的值,例: cout<<q.back();
size() - 队列中元素的个数,例: cout<<q.size();
empty() - 判断队列是否为空,是为ture,否为false 例: q.empty()

queue 队列

```
int main ()
{
    queue<int> que;                          //申明一个int类型的queue变量que
    int sum =0;
    for (int i=1;i<=10;i++) que.push(i);     //将数字1到10入队
    cout<<que.size();                        //输出队中元素个数10
    while (que.empty()==false)               //只要队不为空
    {
        sum += que.front();                  //将队首元素的值累加
        que.pop();                           //队首元素出队
    }
    cout << sum << endl;
}
```

【例1】舞会 nkoi3629

在新年舞会上，n名男士(编号1到n)和m名女士(编号1到m)，各自排成一队。跳舞开始时，依次从男队和女队的队头上各出一人配成舞伴。规定每个舞曲只能有一对跳舞者。一曲结束后，跳舞的一对舞者各自回到自己队伍的末尾。

舞会总共有k个舞曲，问每曲参与跳舞的男士和女士编号是多少?

输入：一行两队的人数和舞曲数量n，m，k

输出：k行，每行两个整数，表示对应舞曲的男士和女士的编号

输入样例:

```
3 4 6
```

输出样例:

```
1 1
2 2
3 3
1 4
2 1
3 2
```

【舞会 参考程序】

```
queue<int>boy;
queue<int>girl;
main()
{
      int m,n,k,i,b,g;
      cin>>n>>m>>k;
      for (i=1;i<=n;i++) boy.push(i);
      for (i=1;i<=m;i++) girl.push(i);
      for (i=1;i<=k;i++)
      {
            b=boy.front();
            g=girl.front();
            boy.pop();
            girl.pop();
            cout<<b<<" "<<g<<endl;
            boy.push(b);
            girl.push(g);
      }
}
```

例2：纸牌游戏 nkoi1917

桌上有一叠纸牌，共n张牌。从位于顶端的纸牌开始从上往下依次编号为1到n。现在反复进行以下操作：把位于顶端的牌扔到，然后把新的位于顶端的牌放到整叠牌的底部。直到只剩下一张牌。

输入n(<=100)，输出每次扔掉的牌的编号以及最后剩下的牌的编号。

样例输入：7

样例输出：1 3 5 7 4 2 6

```
queue<int>q;                                    //申明一个存储整数的队列"q"
int main()
{
      int n,i,x;
      scanf("%d",&n);
      for (i=1;i<=n;i++)q.push(i);              //将n个数字依次加入队列
      while (q.size())                          //当队列不为空q.size()!=0
      {
            printf("%d ",q.front());                  //输出队首元素
            q.pop();                                  //将队首元素删除
             x=q.front();                             //取出队首元素
            q.push(x);                                //将队首元素加入队尾
            q.pop();                                  //删除队首元素
      }
}
```

```
//手工队列版本的代码
int q[201],front,back,i,n;
int main()
{
    cin>>n;
    for(i=1;i<=n;i++)q[i]=i;          //队列赋初值
    front=1;                          //队首指针指向位于顶端的牌的位置
    back=n+1;                         //对尾指针指向下一个空位
    while(front<back)                 //如果队列不为空
    {
        printf("%d ",q[front]);       //输出队首，即扔到最顶端的牌
        front++                       //队首指针指向新的位于顶端的牌
        q[back]=q[front];               //将位于最顶端的牌移到队尾
        back++;                         //对尾指针指向下一个可用空位
        front++;                      //队首指针指向新的位于顶端的牌
    }
   return 0;
}
```

例2：约瑟夫问题 nkoi1700

设有n个人围坐在一个圆桌周围，现从第s个人开始报数，数到第m的人出列，然后从出列的下一个人重新开始报数，数到第m的人又出列，……，如此重复直到所有的人全部出列为止。对于任意给定的n，s和m，求出按出列次序得到的n个人员的顺序表。

```
int main()
{
	int n,s,m;
	cin>>n>>s>>m;
	queue<int> q;
	for(int i=1;i<=n;i++)q.push(i);
	for(int i=1;i<s;i++)
	{
		q.push(q.front());
		q.pop();
	}
	while(q.size())
	{
		for(int i=1;i<m;i++)
		{
			q.push(q.front());
			q.pop();
		}
		cout<<q.front()<<endl;
		q.pop();
	}
}
```