上海工程技术大学

实 验 报 告

实验一　线性表的应用

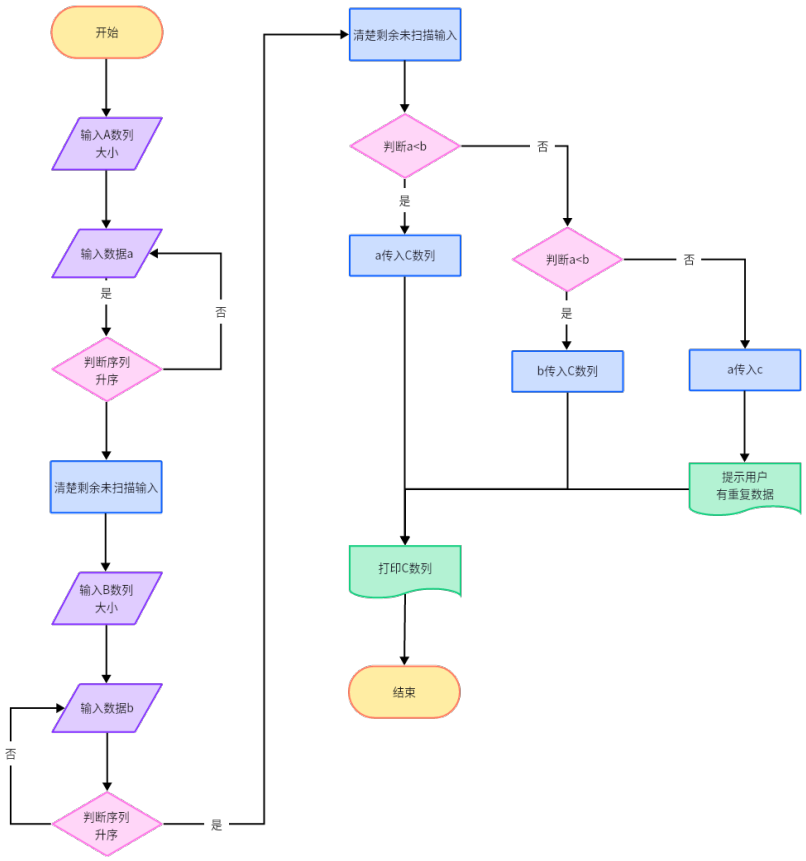
1.　顺序表 题目 1

已知有两个按元素值递增有序的顺序表 A 和 B，设计一个算法将表 A 和表 B 的全部元素归并为一个按元素值非递减有序的顺序表 C。

要求：

从键盘输入顺序表 A 和 B 的各元素，编程实现上述算法，输出顺序表 A、顺序表 B 和顺序表 C 的所有元素值 。

1.1.　流程图



1.2. 代码

```
#include <cstdio>
#define MAXSIZE 100
typedef int ElemType;
typedef struct
{
    ElemType data[MAXSIZE];
    int Len;
}SqList;
void InitList(SqList &sq)
{
    sq={0};
    sq.Len=0;
}
void CreateList(SqList &sq, int n)
{
    ElemType x;
    for (int i = 0; i < n; i++) {
        scanf("%d", &x);
        sq.data[i] = x;
        sq.Len++;
    }
    fflush(stdin);
}
void merge(SqList sqA,SqList sqB,SqList &sqC)
{
    int i=0,j=0,k=0,flag=0;
    while(i<sqA.Len&&j<sqB.Len)
    {
        if(sqA.data[i]<sqB.data[j])
        {
            sqC.data[k]=sqA.data[i];
            i++;
        }
        else if(sqA.data[i]>sqB.data[j])
        {
            sqC.data[k]=sqB.data[j];
            j++;
        }
        else
        {
            sqC.data[k]=sqA.data[i];
            i++;
            j++;
```

```
            flag=1;
        }
        k++;
        sqC.Len++;
    }
    while(i<sqA.Len)
    {
        sqC.data[k]=sqA.data[i];
        i++;
        k++;
        sqC.Len++;
    }
    while(j<sqB.Len)
    {
        sqC.data[k]=sqB.data[j];
        j++;
        k++;
        sqC.Len++;
    }
    if(flag==1)
    {
        printf("Deleted the repeated element\n");
    }
}
void OrderCheck(SqList &sq)
{
    int i=1,j=sq.Len;
    while(i<sq.Len)
    {
        if(sq.data[i-1]>=sq.data[i])
        {
            printf("Not in ascending order\n");
            InitList(sq);
            printf("Enter the elements again:");
            CreateList(sq,j);
            break;
        }
        i++;
    }
}
int main() {
    SqList sqA,sqB,sqC;
    int len;
    InitList(sqA);
```

```
    InitList(sqB);
    InitList(sqC);
    printf("Enter the size of the table A:");
    scanf("%d",&len);
    fflush(stdin);
    printf("Enter the elements of the table A:");
    CreateList(sqA,len);
    OrderCheck(sqA);
    printf("Enter the size of the table B:");
    scanf("%d",&len);
    fflush(stdin);
    printf("Enter the elements of the table B:");
    CreateList(sqB,len);
    OrderCheck(sqB);
    merge(sqA,sqB,sqC);
    for (int i = 0; i < sqC.Len; ++i)
    {
        printf("%5d",sqC.data[i]);
    }
    return 0;
}
```

1.3. 结果

```
Enter the size of the table A:5
Enter the elements of the table A:2 1 7 4 8
Not in ascending order
Enter the elements again:1 2 4 7 8 9
Enter the size of the table B:3
Enter the elements of the table B:2 5 6
Deleted the repeated element
    1    2    4    5    6    7    8
```

2. 顺序表 题目 2

已知线性表 A 按顺序存储，且每个元素都是互不相等的整数。编程实现把所有偶数移到所有的奇数前边的算法。

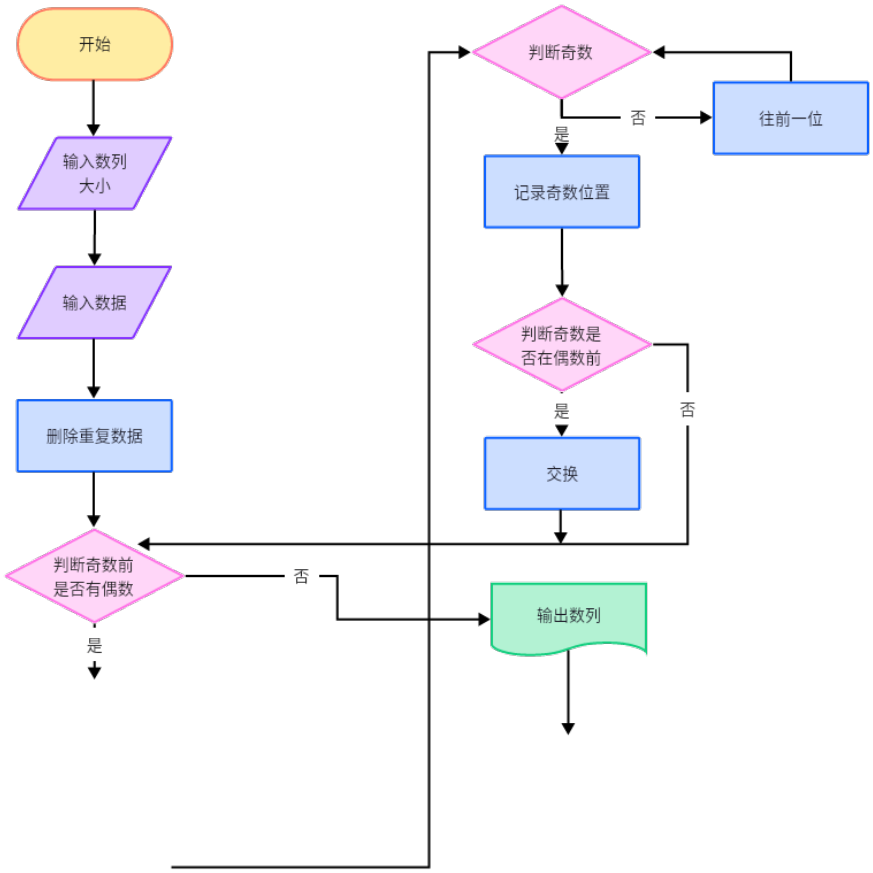
要求：

（1）时间最少，辅助空间最少；

（2）线性表 A 的各元素初始值从键盘输入；

（3）输出结果。

2.1. 流程图



2.2. 代码

```
#include <iostream>
#define MAXSIZE 20
typedef int ElemType;
typedef struct {
    ElemType data[MAXSIZE];
    int len;
}SqList;
void move(SqList &sq) {
    ElemType t;
    int i=0,j=sq.len-1;
    while(i<=j) {
        while(sq.data[i]%2==1) i++;
        while(sq.data[j]%2==0) j--;
        if(i<j) {
            t=sq.data[i];
            sq.data[i]=sq.data[j];
            sq.data[j]=t;
            i++;
            j--;
```

```
        }
    }
}
int main() {
    SqList SqA;
    int num;
    printf("Input the number of SqA:");
    scanf("%d",&num);
    printf("Input the data of SqA:");
    for (int i = 0; i < num; ++i) {
        scanf("%d",&SqA.data[i]);
        for (int j = 0; j < i; ++j) {
            if(SqA.data[i]==SqA.data[j]){
                printf("anything repeated and deleted\n");
                i--;
                num--;
            }
        }
    }
    SqA.len=num;
    move(SqA);
    for (int i = 0; i < num; ++i) {
        printf("%5d",SqA.data[i]);
    }
    return 0;
}
```

2.3. 结果

```
Input the number of SqA:6
Input the data of SqA:1 3 5 2 3 4
anything repeated and deleted
    1    3    5    2    4
```

3. 链表 题目 1

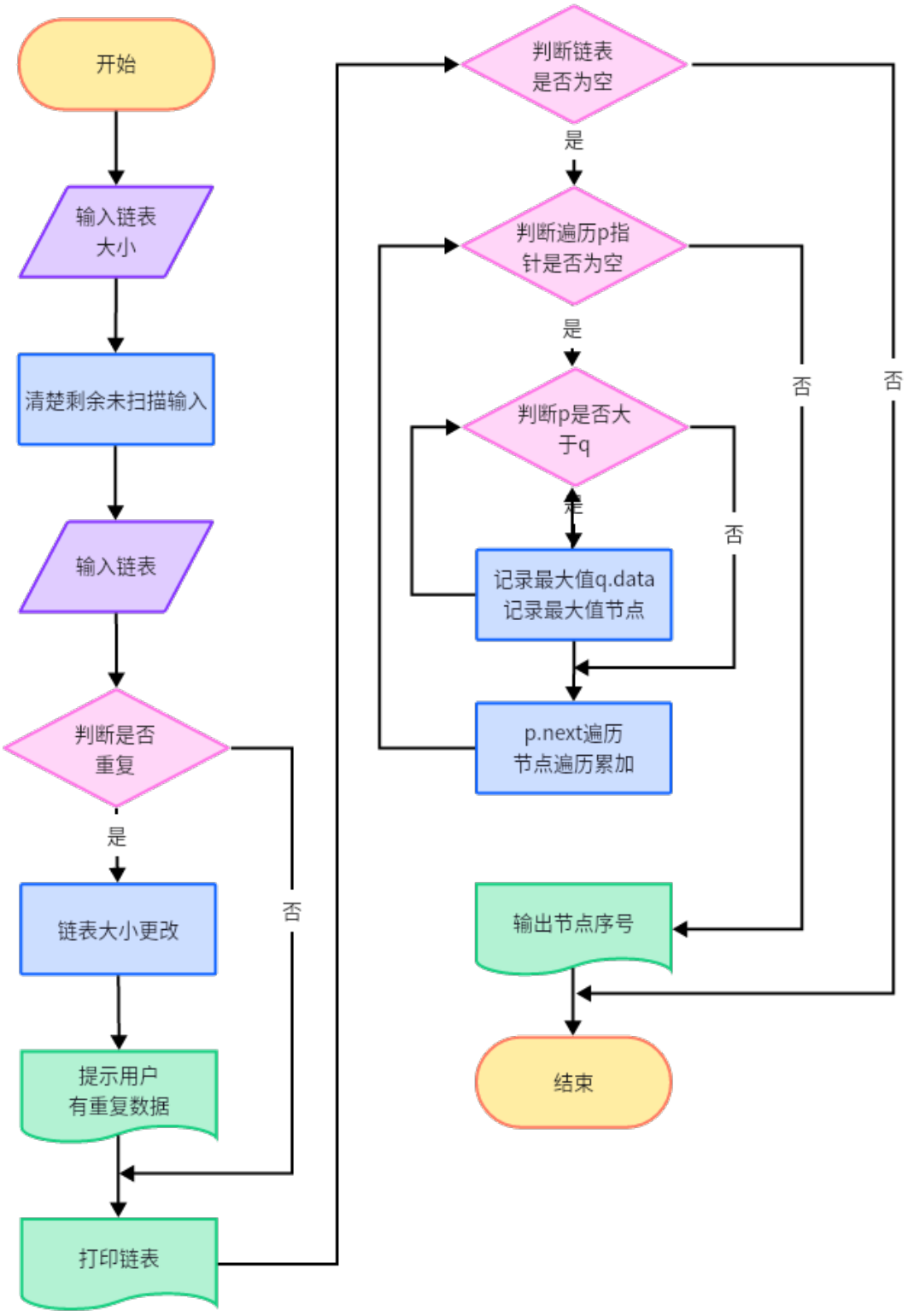
设单链表的数据为互不相等的整数，建立一个单链表，并设计一个算法,找出单链表中元素值最大的结点。

要求：

（1）单链表的数据从键盘输入；

（2）输出单链表所有结点的数据和最大值结点序号。

3.1. 流程图



3.2. 代码

```
#include <cstdio>
#include <cstdlib>
#define N 1000
```

```
typedef int ElementType;
typedef struct LNode{
    ElementType data;
    struct LNode *Next;
}LNode,*LinkList;
void CreateLink(LinkList &h, ElementType a[], int n)
{
   LNode *s, *r; int i;
   h = (LNode *)malloc(sizeof(LNode));
   r = h;
   for (i = 0; i < n; i++)
   {
      s = (LNode *)malloc(sizeof(LNode));
      s->data = a[i];
      r->Next = s; r = s;
   }
   r->Next = NULL;
}
int MaxNode(LinkList h){
    int j,k;
    LNode *p,*q;
    if(h->Next==NULL){
        return 0;
    }
    q=h->Next;
    p=q->Next;
    k=1;
    j=2;
    while(p!=NULL){
        if(p->data>q->data){
            q=p;
            k=j;
        }
        p=p->Next;
        j++;
    }
    return k;
}
int Print(LinkList h){
    LNode *q;
    if(h->Next==NULL){
        return 0;
    }
    q=h->Next;
```

```
    while(q!=NULL){
        printf("%d ",q->data);
        q=q->Next;
    }
}
int main(){
    LinkList head;
    ElementType a[N];
    int i,k,num,flag=0;
    printf("Enter the size of the list:\n");
    scanf("%d",&num);
    fflush(stdin);
    printf("Input a list of number:\n");
    for(i = 0; i < num; i++){
        scanf("%d",&a[i]);
        for (int j = 0; j < i; ++j) {
            if(a[i]==a[j]){
                flag = 1;
                i--;
                num--;
            }
        }
    }
    if(flag == 1) printf("Anything repeated and deleted\n");
    CreateLink(head,a,num);
    printf("Elements in list:\n");
    Print(head);
    printf("\n");
    printf("The No. of the max element\n");
    k=MaxNode(head);
    printf("%d",k);
    return 0;
}
```

3.3. 结果

```
Enter the size of the list:
5 1 2
Input a list of number:
3 5 6 9 11
Elements in list:
3 5 6 9 11
The No. of the max element
5
```

4. 链表 题目 2

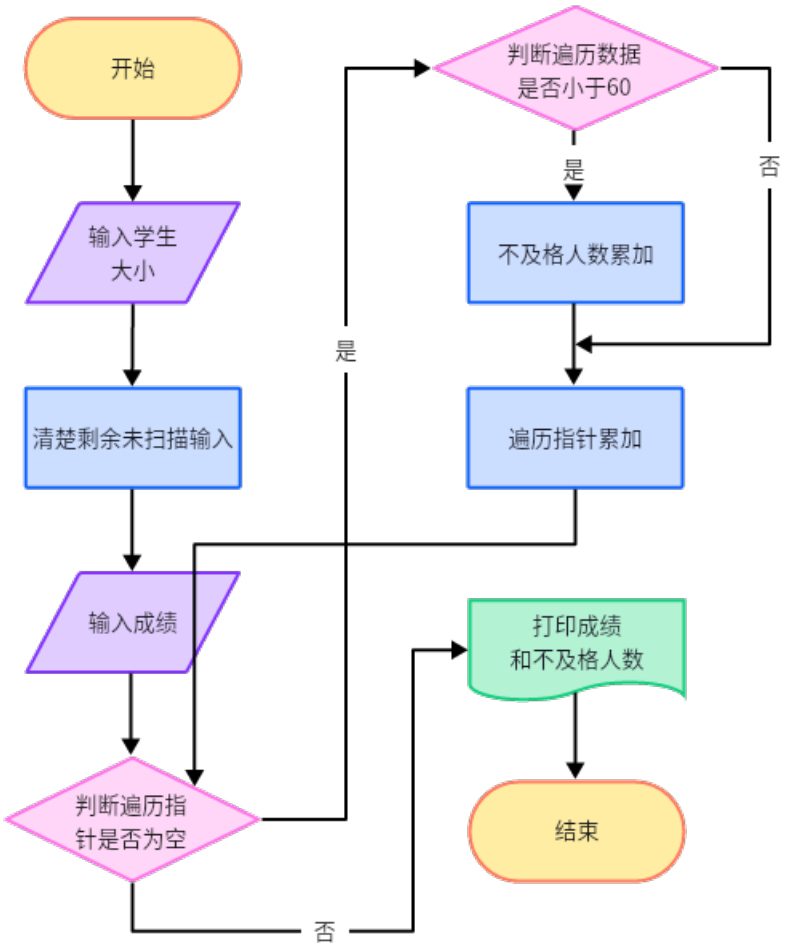
设计算法，根据输入的学生人数和成绩建立一个单链表，并累计成绩不及格的人数。

要求：

（1）学生人数和成绩均从键盘输入；

（2）输出所有学生的成绩和不及格的人数。

4.1. 流程图



4.2. 代码

```
#include <cstdio>
#include <malloc.h>
typedef int ElemType;
typedef struct node
{
    ElemType data;
    struct node *next;
} StudNode, *StudLink;
```

```
void create(StudLink &sl)
{
    int i, n, score;
    StudNode *s, *r;
    sl = (StudNode*)malloc(sizeof(StudNode));
    r = sl;
    printf("Input the number of students:\n");
    scanf("%d", &n);
    fflush(stdin);
    printf("Input scores of students:\n");
    for (i = 0; i < n; i++){
        s = (StudNode*)malloc(sizeof(StudNode));
        scanf("%d", &score);
        s->data = score;
        r->next = s;
        r = s;
    }
    r->next = NULL;
}
int output(StudLink sl)
{
    StudNode *q;
    if (sl->next == NULL) return 0;
    q = sl->next;
    while (q != NULL)
    {
        printf("%d\t", q->data);
        q = q->next;
    }
}
int count(StudLink sl)
{
    int n = 0;
    StudNode *p = sl->next;
    while (p != NULL)
    {
        if (p->data < 60) n++;
        p = p->next;
    }
    return n;
}
int main()
{
```

```
    int n;
    StudLink h;
    create(h);
    n = count(h);
    printf("The list of their scores:\n");
    output(h);
    printf("\nThe number of fail students:\n%d", n);
    return 0;
}
```

4.3. 结果

```
Input the number of students:
5
Input scores of students:
98 97 96 12 40 54
The list of their scores:
98      97      96      12      40
The number of fail students:
2
```

5. 栈 题目 1

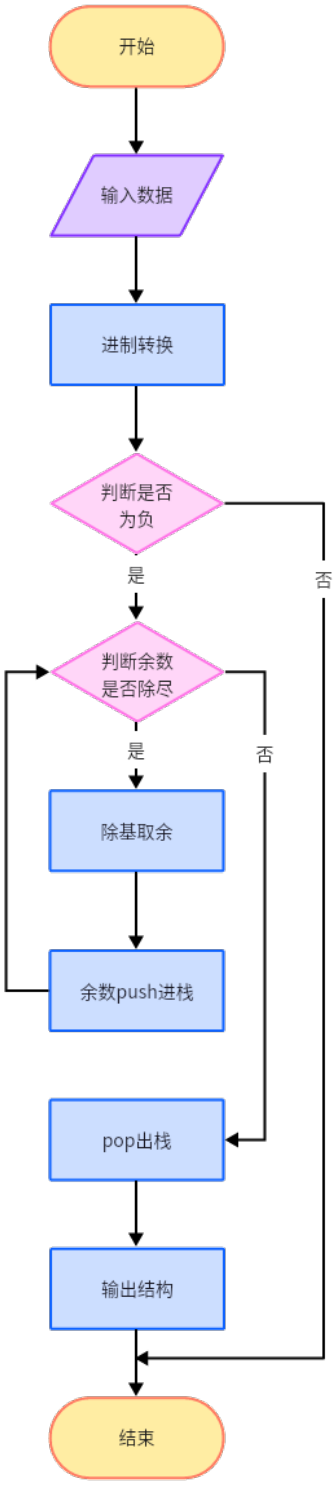
编写一个算法，将非负的十进制整数转换为其他进制的数输出，10 及其以上的数字从‘A’开始的字母表示。

要求:

1）采用顺序栈实现算法；

2）从键盘输入一个十进制的数，输出相应的八进制数和十六进制数。

5.1. 流程图



5.2. 代码

```
#include <cstdio>
#define MaxSize 100
typedef char ElemType;
typedef struct{
    ElemType data[MaxSize];
    int top;
```

```
} SqStack;
int trans(int num,int sys,char string[]){
    SqStack str;
    char ch;
    int r,i=0;
    str.top=-1;
    if(num<=-1){
        printf("Negative number! ");
        return 0;
    }
    while(num!=0){
        r=num%sys;
        ch=r+(r<10?'0':'A'-10);
        str.top++;
        str.data[str.top]=ch;
        num/=sys;
    }
    while(str.top!=-1){
        string[i++]=str.data[str.top];
        str.top--;
    }
    string[i]='\0';
    return 1;
}
int main(){
    char str[10];
    int num,t;
    printf("Input the number and non-negative:");
    scanf("%d",&num);
    t=trans(num,8,str);
    if(t==0){
        printf("ERROR\n");
    }else{
        printf("OCT:%s\n",str);
    }
    t=trans(num,16,str);
    if(t==0){
        printf("ERROR\n");
    }
    else{
        printf("HEX:%s\n",str);
    }
    return 0;
}
```

5.3. 结果

```
Input the number and non-negative:-5
Negative number! ERROR
Negative number! ERROR
Input the number and non-negative:2023
OCT:3747
HEX:7E7
```

6. 栈 题目 2

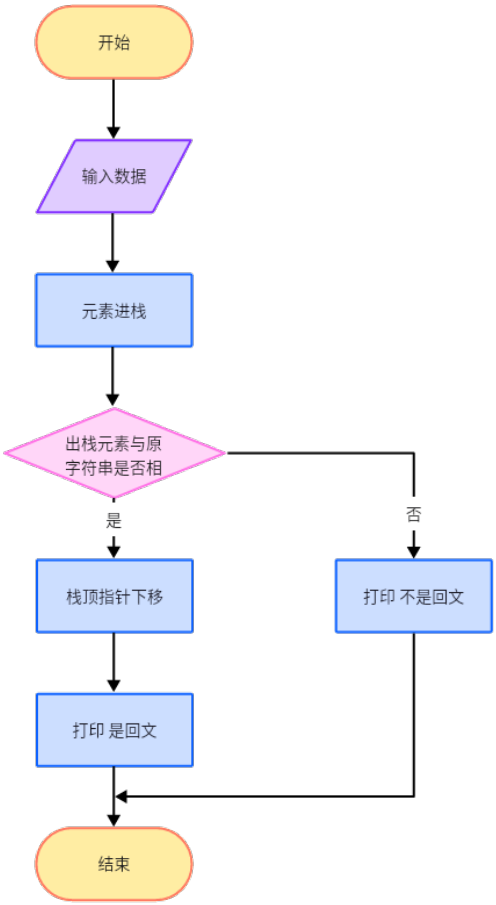
回文指的是一个字符串从前面读和从后面读都一样，编写一个算法判断一个字符串是否为回文。

要求：

1）采用链栈实现算法；

2）从键盘输入一个字符串，输出判断结果。

6.1. 流程图



6.2. 代码

```
#include <cstdio>
#include <cstdlib>
#define ElementType char
typedef struct lineStack{
    ElementType data;
    struct lineStack * next;
}lineStack;
lineStack* push(lineStack * stack,char a){
    lineStack * line=(lineStack*)malloc(sizeof(lineStack));
    line->data=a;
    line->next=stack;
    stack=line;
    return stack;
}
int palindrome(ElementType X[]){
    lineStack * stack=NULL;
    int i;
    for(i=0;X[i]!='\0';i++){
        stack=push(stack,X[i]);
    }
    char str;
    lineStack *p;
    for(i=0;X[i]!='\0'&&stack!=NULL;i++){
        p=stack;
        str=p->data;
        stack=stack->next;
        free(p);
        if(X[i]!=str){
            return 0;
        }
    }
    return 1;
}
int main(){
    char ch[20];
    printf("Input a string:\n");
    scanf("%s",&ch);
    int i=palindrome(ch);
    if(i==1){
        printf("%s is palindrome\n",ch);
    }
    else{
        printf("%s is not palindrome\n",ch);
```

```
    }
    return 0;
}
```

6.3. 结果

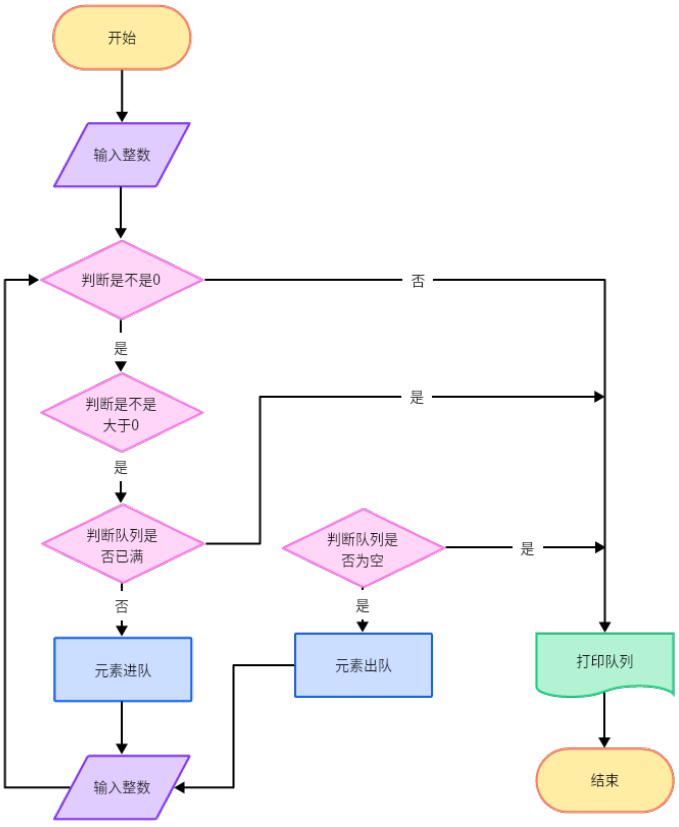
```
Input a string:
abba
abba is palindrome
```

```
Input a string:
abcd
abcd is not palindrome
```

7. 队列 题目 1

从键盘输入一字符串，试编程实现：数字字符进队；当遇见负数时，将队首元素出队；当遇见正数时，将队首元素出队；当遇见零时，将输入结束；其他字符忽略。最后输出队列中的所有字符。要求采用循环队列存储结构。

7.1. 流程图



7.2. 代码

```
#include <cstdio>
#define QueueSize 10
typedef int ElementType;
```

```
typedef struct{
    ElementType Data[QueueSize];
    int front;
    int rear;
}SqQueue;
void PrintQueue(SqQueue qu) {
    int i = qu.front;
    printf("Queue:");
    while (i != qu.rear) {
        printf("%d ", qu.Data[i]);
        i = (i + 1) % QueueSize;
    }
    printf("\n");
}
int main(){
    ElementType a,x;
    SqQueue qu;
    qu.rear=qu.front=0;
    printf("Input queue(end with 0, InQueue with positive, DeQueue with
negative)\n");
    scanf("%d",&a);
    while(a!=0){
        if(a>0){
            if((qu.rear+1)%QueueSize==qu.front){
                printf("Full Queue!\n");
                PrintQueue(qu);
                return 0;
            }
            qu.Data[qu.rear]=a;
            qu.rear=(qu.rear+1)%QueueSize;
        }else{
            if(qu.rear==qu.front){
                printf("NUll Queue!\n");
                return 0;
            }
            x=qu.Data[qu.front];
            qu.front=(qu.front+1)%QueueSize;
            printf("%dDeQueue\n",x);
        }
        scanf("%d",&a);
    }
    PrintQueue(qu);
    return 0;
}
```

7.3. 结果

```
Input queue(end with 0, InQueue with positive, DeQueue with negative)
2
1
3
4
5
6
-1
2DeQueue
-2
1DeQueue
-3
3DeQueue
0
Queue:4 5 6
```

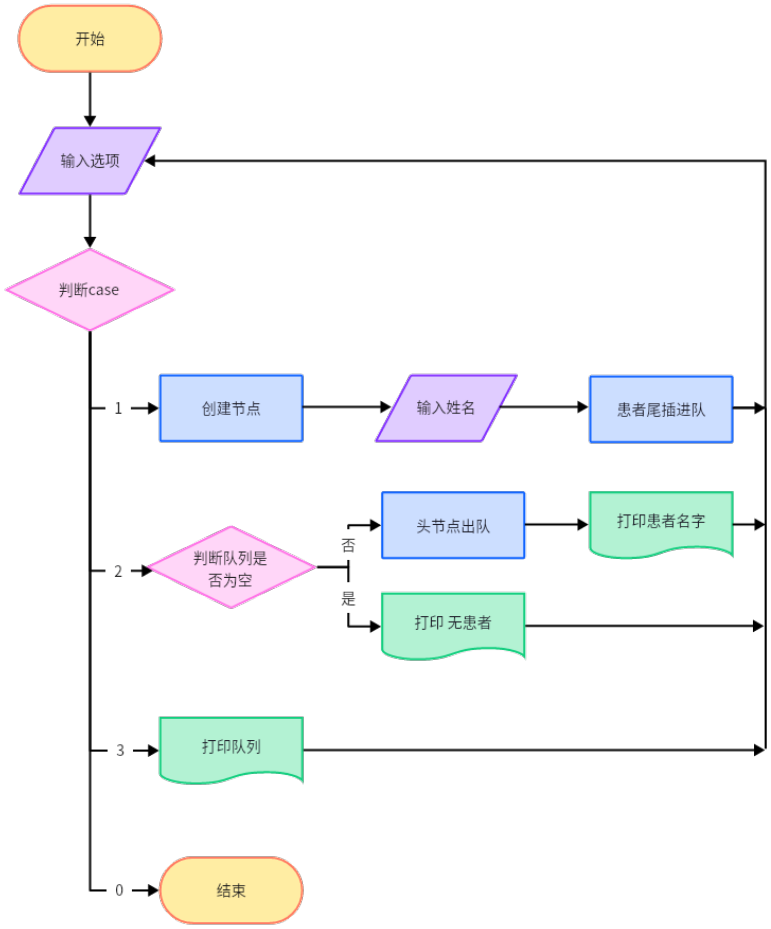
```
Full Queue!
Queue:1 1 1 1 1 1 1 1
```

8. 队列 题目 2

设计一个程序，反映病人到医院看病、排队看医生的情况。

要求：采用链队列存储结构

8.1. 流程图



8.2. 代码

```
#include <cstdio>
#include <cstring>
#include <malloc.h>

typedef char ElemType;
typedef  struct  QNode
{
    ElemType data[15];
    struct QNode  *next;
} QNode,*QueuePtr;
typedef  struct{
    QueuePtr  front;
    QueuePtr  rear;
}LinkQueue;
int main()
{
    int choice,flag=1;
    LinkQueue lq;
    QNode *s,*p;
    char name[15];
```

```
    lq.front=( QueuePtr)malloc(sizeof(QNode));
    lq.front->next=NULL;
    lq.rear=lq.front;
    while(flag==1)
    {
        printf("1.排队 2.就诊 3.查看排队情况 0.下班\n");
        printf("请选择：");
        scanf("%d",&choice);
        switch(choice)
        {
            case 0:
                if (lq.front!=lq.rear) printf(">>下班了，请排队的患者
改天再来就医\n");
                flag=0; break;
            case 1:
                printf("请输入患者姓名：");
                scanf("%s",name);
                s=( QueuePtr)malloc(sizeof(QNode));
                strcpy(s->data,name);
                s->next=NULL;
                lq.rear->next=s; lq.rear=s;
                break;
            case 2:
                if (lq.front==lq.rear)
                    printf(">>没有排队的患者\n");
                else
                {
                    s=lq.front->next;
                    lq.front->next=s->next;
                    if (lq.rear==s) lq.rear=lq.front;
                    printf("%>>s 请就诊\n",s->data);
                    free(s);
                }
                break;
            case 3:
                if (lq.front==lq.rear)
                    printf(">>没有排队的患者\n");
                else
                {
                    p=lq.front->next;
                    printf(">>排队的患者：");
                    while(p!=NULL)
                    {
                        printf("%s    ",p->data);
```

```
                p=p->next;
            }
            printf("\n");
        }
        break;
    }
  }
  return 0;
}
```

8.3. 结果

```
1.排队 2.就诊 3.查看排队情况 0.下班
请选择：1
请输入患者姓名：zny
1.排队 2.就诊 3.查看排队情况 0.下班
请选择：1
请输入患者姓名：data
1.排队 2.就诊 3.查看排队情况 0.下班
请选择：3
>>排队的患者：zny    data
1.排队 2.就诊 3.查看排队情况 0.下班
请选择：0
>>下班了，请排队的患者改天再来就医
```

9. 实验小结

在实验过程中，加深了对线性表的理解，熟悉了顺序的创建和插入算法的程序编写；熟悉了创建和插入单链表的算法的程序编写；熟悉了栈创建，入栈，出栈算法程序的编写。；熟悉了队列的创建、出队、入队的程序的编写。

在编写过程中，由于是第一次真实使用 CLion 进行编写，与 VS 的区别还是比较大的，整体的编写体验比 VS 好了不少，但是由于缺少中文适配，在初期使用过程中有一些不适应。例如，UTF-8 和 GBK 的区别，在后来发现 GBK 才能适配中文的输出，UTF-8 里的中文无法输出；CLion 需要进行 CMake 的编写，添加的 cpp 要自行配置编译器，运行和调试会默认使用 CLion 自己的控制台和终端。在调试过程中也写了很多的校验，防止用户的错误输入：添加升序校验函数，确保代码的正常运行；在每次输入前，添加 fflush 函数，以免上一次的错误输入影响这次的输入；在函数引用中，通过&对数据进行更改，但是可以 return 一个 bool 变量，来返回函数状态，这一次用的 1/0 表示状态，在以后会使用 bool 函数提升效率。