lab6：用C语言实现lab2~5

lab2:gcd算法

首先判断输入的两个数的大小，将大的数设置为a，小的数设置成b,

用辗转相除法：

```
    while(b!=0){
        c=a%b; a=b;  b=c;
    }
```

a即为最大公约数。

lab2链表排序

这里选择冒泡排序。

链表定义：

```
typedef struct link_list{
    int value;
    struct link_list *next;
}LINK;
```

链表创建：

```
    head->next=p;
    printf("How many numbers do you want to input:");
    scanf("%d",&n);
    printf("Please input numbers:\n");
    for(i=0;i<n;i++)
    {
        scanf("%d",&p->value);
        if(i==n-1){
            p->next=NULL;
            break;
        }
        p->next=(struct link_list*)malloc(sizeof(struct link_list));//申请下一个节点
        p=p->next;//将该节点和下一个节点连起来
    }
```

冒泡排序：

```
    while(pf->next!=NULL){
        pb = pf->next;//pb从基准点的下一个节点开始
        while(pb != NULL) {
            if(pf->value > pb->value) {
                temp = *pf;
                *pf = *pb;
                *pb = temp;
                temp.next = pf->next;
                pf->next = pb->next;
                pb->next = temp.next;
            }
            pb = pb->next;
        }
        pf = pf->next;

    }
```

输出结果：

```
    p=head->next;//由于第一个循环已经将链表移到末尾，所以这里要将链表移到首结点开始打印
    for(i=0;i<n;i++)
    {
        printf("%d ",p->value);
        p=p->next;
    }
```

lab4:NIM游戏

用变量a1,a2,a3来储存A,B,C行剩余的石头数量。用flag标记当前轮到哪个player，首先打印当前各行石头数量信息，之后用scanf获取player的操作信息，并且加入错误判断，当A,B,C行石头数量均为0时游戏结束，打印获胜信息,跳出循环。

①判断游戏是否结束

```
        if(a1==0&&a2==0&&a3==0){
            break;
        }
```

②打印当前信息

```
        printf("Row A:");
        for(int i=0;i<a1;i++){
            printf("o");
        }
        printf("\n");
        printf("Row B:");
        for(int i=0;i<a2;i++){
            printf("o");
        }
        printf("\n");
        printf("Row C:");
        for(int i=0;i<a3;i++){
            printf("o");
        }
        printf("\n");
```

③scanf获取player操作

```
label1:     if(flag==0){
            printf("Player 1, choose a row and number of rocks:");
        }
        if(flag==1){
            printf("Player 2, choose a row and number of rocks:");
        }
        getchar();
        scanf("%c%d",&a4,&a5);
        printf("\n");
```

④判断输入是否合法并且处理输入

```
label1:     if(flag==0){
            printf("Player 1, choose a row and number of rocks:");
        }
        if(flag==1){
            printf("Player 2, choose a row and number of rocks:");
        }
        getchar();
        scanf("%c%d",&a4,&a5);
        printf("\n");

        if(a4!='A'&&a4!='B'&&a4!='C'){
            printf("Invalid move. Try again.\n");
            goto label1;
        }
        if(a4=='A'){
            if(a5>a1){
                printf("Invalid move. Try again.\n");
                goto label1;
            }
            else{
                a1=a1-a5;
            }
        }
        if(a4=='B'){
            if(a5>a2){
                printf("Invalid move. Try again.\n");
                goto label1;
            }
            else{
                a2=a2-a5;
            }
        }
        if(a4=='C'){
            if(a5>a3){
                printf("Invalid move. Try again.\n");
                goto label1;
            }
            else{
                a3=a3-a5;
            }
        }
```

⑤转换player轮次

```
        if(flag==0){
            flag=1;
        }
        else{
            flag=0;
        }
```

⑥输出获胜信息

```
    if(flag==0){
        printf("Player 1 Wins.");
    }
    else{
        printf("Player 2 Wins.");
    }
```

lab5:中断

运用#include <conio.h>库中的kbhit() 函数来判断是否有输入，如果有输入，将其存进ch里面并且判断是否是数字。如果没有输出，一直输出"ICS2020 "，用Sleep来控制输出速度。

```
    char ch;
while(1)
{
printf("ICS2020 ");

if( kbhit() )
{
ch = getch();
if(
ch=='0'||ch=='1'||ch=='2'||ch=='3'||ch=='4'||ch=='5'||ch=='6'||ch=='7'||ch=='8'||ch=='9'
){
    printf("%c is a decimal digit",ch);
}
else{
    printf("%c is not a decimal digit",ch);
}
}
Sleep(500);
}
```