```
//第6.1 数制转换
typedef struct
{
    int elem[MAXSIZE];
    int top;
}SqStack;

int GetTop(SqStack S,int *e);
int Pop(SqStack *S,int *e);
void Init(SqStack *S);
int Push(SqStack *S,int e);
void Coversation(int N,int d, SqStack *S);
void Print(SqStack S);
int _tmain(int argc, _TCHAR* argv[])
{
    int N=1348;
    int d = 8;
    SqStack S;
    Init(&S);
    Coversation(N,d,&S);
    Print(S);
    return 0;
}

void Init(SqStack *S)
{
    S->top = -1;
}
int GetTop(SqStack S,int *e)
{
    if(S.top == -1)
        return 0;
```

```
    if(S.top>-1)
        *e = S.elem[S.top];
    return 1;
}
int Push(SqStack *S,int e)
{
    if(S->top == MAXSIZE-1)
        return 0;
    S->elem[++S->top]=e;
    return 1;
}
int Pop(SqStack *S,int *e)
{
    if(S->top == -1)
        return 0;
    *e =S->elem[S->top--];
    return 1;
}

void Coversation(int N,int d,SqStack *S)
{
    while(N)
    {
        Push(S,N%d);
        N=N/d;
    }
}
void Print(SqStack S)
{
    int s=0;

    while(S.top>=0)
```

```
    {
        int e;
        Pop(&S,&e);
        s = s*10 +e;
    }
    printf("\n%d\n",s);
}

//6.2 括号匹配
typedef struct lnode
{
    char data;
    struct lnode *next;
}SNode,*SLink;
void Init(SLink *LS);
int Push(SLink *LS,char e);
int Pop(SLink *LS,char *e);
int GetTop(SLink S,char *e);
int BracketMatch(char exp[]);
int EmptyStack(SLink S);
int charMatch(char c1,char c2 );

int _tmain(int argc, _TCHAR* argv[])
{
    char exp[]="[(1+2)*(2+3)]*[5-1]#";
    int i = BracketMatch(exp);

    return 0;
}
int isEmptyStack(SLink S)
{
    return S == NULL;
```

```
}
int GetTop(SLink S,char *e)
{
    if(S==NULL)
        return 0;
    *e = S->data;
    return 1;
}
int charMatch(char c1,char c2 )
{
    if(c1 == '(' && c2==')')
        return 1;
    if(c1 == '{' && c2 == '}')
        return 1;
    if(c1=='[' && c2 == ']')
        return 1;
    return 0;
}
int BracketMatch(char exp[])
{
    char *p = exp;
    SLink S;
    Init(&S);

    while(*p)
    {
        if(*p == '(' || *p == '[' || *p == '{')
            Push(&S,*p);
        else if(*p == ')' || *p == ']' || *p == '}')
        {
            char e;
            if(isEmptyStack(S)==0)
```

```
            {
                GetTop(S,&e);
                if(charMatch(e,*p))
                    Pop(&S,&e);
                else
                    break;
            }
            else if(isEmptyStack(S))
                break;
        }
        p++;
    }
    if(isEmptyStack(S) && *p == '\0')
        return 1;
    else
        return 0;
}
int Pop(SLink *LS,char *e)
{
    if(*LS == NULL)
        return 0;
    *e = (*LS)->data;
    *LS=(*LS)->next;
    return 1;
}
void Init(SLink *LS)
{
    *LS = NULL;
}
int Push(SLink *LS,char e)
{
    SLink s = NULL;
```

```
    s = new SNode;
    if(!s)
        return 0;
    s->data = e;
    s->next = *LS;
    *LS = s;
    return 1;
}
//6.3 双栈保存奇数偶数
typedef struct
{
    int elem[MAXSIZE];
    int top1;
    int top2;
}DStack;
int Push(DStack *LS,int e, int flag);
void Init(DStack *LS);
int Pop(DStack *LS,int *e,int flag);
void Print(DStack S);
void ReadToSave(DStack *LS);
int _tmain(int argc, _TCHAR* argv[])
{
    DStack S;
    Init(&S);
    ReadToSave(&S);
    Print(S);

    return 0;
}
void ReadToSave(DStack *LS)
{
    int a;
```

```
    while(scanf("%d",&a))
    {
        if(a%2==0)
            Push(LS,a,2);
        else
            Push(LS,a,1);
    }
}
void Print(DStack S)
{
    while(S.top1!=-1)
        printf("%d\t",S.elem[S.top1--]);
    printf("\n\n");
    while(S.top2!=MAXSIZE)
        printf("%d\t",S.elem[S.top2++]);
}
int Pop(DStack *LS,int *e,int flag)
{
    if(flag !=1 || flag !=2)
        return 0;
    if(LS->top1 == -1 && flag == 1)
        return 0;
    else if(LS->top2 == MAXSIZE && flag == 2)
        return 0;
    if(flag == 1)
        *e = LS->elem[LS->top1--];
    else if(flag == 2)
        *e = LS->elem[LS->top2++];
    return 1;
}

void Init(DStack *LS)
```

```
{
    LS->top1 = -1;
    LS->top2 = MAXSIZE;
}
int Push(DStack *LS,int e, int flag)
{
    if(LS->top1+1 == LS->top2)
        return 0;
    if(flag == 1)
        LS->elem[++LS->top1] = e;
    else if(flag == 2)
        LS->elem[--LS->top2] = e;
    else
        return 0;
    return 1;
}
```