```
#include<stdio.h>
#include<stdlib.h>
void push(int x);
void pop();
void stack_top();
int x,y,ch=1,value,z,top=-1,stack[10];
int main(void)
{
    while(1)
    {
        printf("\nSTACK MANIPULATION\n");
        printf("1.push\n2.pop\n3.top of
stack\n4.exit\n");
        printf("Enter the choice:");
        scanf("%d",&ch);
        switch(ch)
        {
        case 1:printf("Enter value to be pushed
in:");
              scanf("%d",&value);
```

```
            push(value);
            break;
        case 2:pop();
            break;
        case 3:stack_top();
            break;
        case 4:exit(0);
        default:printf("wrong choice...Retry\n");
        }
        }
        }

void push(int x)
{int i;
    if(top==9)
    printf("stack overflows\n");
    else
    {
        top++;
        stack[top]=x;
```

```
        printf("\n The stack is\n");
       for(i=top;i>=0;i--)
              printf("%d\n",stack[i]);
        }
        }

void pop()
{
    int i;
    if(top==-1)
    printf("stack underflows");
    else
    {
        printf("The value %d is
popped\n",stack[top]);
        top--;
        }
        printf("The stack is\n");
      for(i=top;i>=0;i--)
              printf("%d\t",stack[i]);
```

```
        }
```

```
void stack_top()
{
    if(top==-1)
    printf("stack underflows");
    else
    {
        int y=stack[top];
        printf("The top of the stack is %d\n",y);
        }
        }
```

Balanced parenthesis

```
#include<stdio.h>
#include<string.h>
int main(void)
```

```
{
    char stack[10],exp[10];
    int i,len,top=-1,diff;
    printf("Enter the expression");
    scanf("%s",exp);
    len=strlen(exp);
    for(i=0;i<len;i++)
    {
        if(exp[i]=='('||exp[i]=='[' )
        {
            top=top+1;
            stack[top]=exp[i];
        }
        else if (exp[i]==')'||exp[i]==']' )
        {
            if (top==-1)            //a+b)
            {printf("Missing Open Bracket");
            return 0;
        }
            else                    //(a+b]
```

```
            {
                diff=exp[i]-stack[top];

            printf("diff=%d\n",diff);
            if(diff==1 || diff==2)
            {
            printf("Valid Brackets\n"); //(a+b)
            top=top-1;
          }
            else //(a+b]
            {
            printf("Invalid Brackets");
            return 0;
      }
            }
      }
      else
      continue;
}
if(exp[i]=='\0'&& top!=-1)
```

```
        printf("Missing Close Bracket"); //(a+b
/*      else      //a+b
        printf("Valid Expression");*/
        return 0;
}
```