DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

PROGRAMMING LABORATORY (CSE 351)

ASSIGNMENT 4

Asmit De
10/CSE/53

Date: 15.09.2011

Program 1: Infix to Postfix Conversion

Source Code –

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
#define MAX 20

char stack[MAX];
int top = -1;

void push(char x)
{
    if(top == MAX-1)
    {
       printf("\nError: Stack overflow...");
       getch();
       exit(1);
    }

    stack[++top] = x;
    return;
}

char pop()
{
    if(top < 0)
    {
       printf("\nError: Stack underflow...");
       getch();
       exit(1);
    }

    return stack[top--];
}

int main()
{
    char c, infix[20], postfix[20];
    int i = 0, p = 0;

    printf("Enter infix expression: ");
    gets(infix);

    while((c = infix[i++]) != '\0')
    {
       switch(c)
       {
             case '^':
              case '(':
                     push(c);
                     break;

              case ')':
                     while(stack[top] != '(')
                            postfix[p++] = pop();
                     pop();
                     break;

              case '/':
              case '*':
                     while(!(stack[top] == '+' || stack[top] == '-' || stack[top] ==
                     '(' || top < 0))
                            postfix[p++] = pop();
                     push(c);
                     break;

              case '+':
              case '-':
                     while(!(stack[top] == '(' || top < 0))
                            postfix[p++] = pop();
                     push(c);
                     break;

              default:
                     postfix[p++] = c;
       }
    }

    while(top >= 0)
       postfix[p++] = pop();
    postfix[p] = '\0';

    printf("\nPostfix expression: %s\n", postfix);

    getch();
    return 0;
}
```

Output–

```
Enter infix expression: A*B^C^D+E/(F+G)-H

Postfix expression: ABCD^^*EFG+/+H-
```

Program 2: Evaluation of Postfix Expression

Source Code –

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
#include <math.h>
#define MAX 20

float stack[MAX];
int top = -1;

void push(float x)
{
    if(top == MAX - 1)
    {
       printf("\nError: Stack overflow...");
       getch();
       exit(1);
    }

    stack[++top] = x;
    return;
}

float pop()
{
    if(top < 0)
    {
       printf("\nError: Stack underflow...");
       getch();
       exit(1);
    }

    return stack[top--];
}

int main()
{
    char c, postfix[20];
    int i = 0;
    float val, a, b, result;

    printf("Enter postfix expression: ");
    gets(postfix);

    while((c = postfix[i++]) != '\0')
    {
       switch(c)
       {
              case '^':
                     b = pop();
                     a = pop();
                     result = pow(a, b);
                     push(result);
                     break;

              case '/':
                     b = pop();
                     a = pop();
                     result = a / b;
                     push(result);
                     break;

              case '*':
                    b = pop();
                    a = pop();
                    result = a * b;
                    push(result);
                    break;

             case '+':
                    b = pop();
                    a = pop();
                    result = a + b;
                    push(result);
                    break;

             case '-':
                    b = pop();
                    a = pop();
                    result = a - b;
                    push(result);
                    break;

             default:
                    printf("Enter the value of %c: ", c);
                    scanf("%f", &val);
                    push(val);
       }
    }

    printf("\nResult = %f\n", result);

    getch();
    return 0;
}
```

Output–

```
Enter postfix expression: ABCD+*EF*/+
Enter the value of A: 5
Enter the value of B: 12
Enter the value of C: 2
Enter the value of D: 1
Enter the value of E: 3
Enter the value of F: 1

Result = 17.000000
```

Program 3: Implementation of a Stack using Array

Source Code –

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
#define MAX 50

int stack[MAX], top = -1;

int push(int data)
{
    if(top == MAX - 1)
       return 1;

    stack[++top] = data;
    return 0;
}

int pop()
{
    if(top < 0)
    {
       printf("\nError: Stack underflow...");
       return NULL;
    }

    return stack[top--];
}

int main()
{
    char c;
    int i, data, d;

    while(1)
    {
       system("cls");
       puts("Choose an operation: ");
       puts("1\tDisplay contents of stack");
       puts("2\tPush data to stack");
       puts("3\tPop data from stack");
       puts("4\tExit");
       puts("\nEnter choice...");

       c = getch();
       fflush(stdin);

       switch(c)
       {
           case '1':
              system("cls");

              printf("Stack contents:");
              for(i = 0; i <= top; i++)
                  printf("\n%d", stack[i]);

              printf("\n\nPress any key to return to menu...");
              getch();
              break;

           case '2':
              system("cls");

              printf("Enter the data to be pushed to stack: ");
```

```
             scanf("%d", &data);

             if(!push(data))
                 printf("\nData pushed successfully...Press any key to return to
                 menu...");
             else
                 printf("\nError: Stack overflow...Press any key to return to
                 menu...");
             getch();
             break;

          case '3':
             system(“cls”);

             printf("Popped value: %d \nPress any key to return to menu...",
             pop());
             getch();
             break;

          case '4':
             exit(0);

          default:
             system("cls");

             printf("Invalid input...Press any key to return to menu...");
             getch();
       }
    }
}
```

Program 4: Implementation of a Stack using Linked List

Source Code –

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>

typedef struct node
{
    int data;
    struct node *next;
}NODE;

NODE *top = NULL, *temp;

int push(int num)
{
    temp = (NODE*)malloc(sizeof(NODE));
    if(temp == NULL)
       return 1;

    temp->data = num;
    temp->next = top;
    top = temp;
    return 0;
}

int pop()
{
    int value;
    if(top == NULL)
    {
       printf("\nEmpty Stack\n");
       return NULL;
    }

    temp = top;
    value = temp->data;
    top = temp->next;
    free(temp);
    return value;
}

int main()
{
    char c;
    int data;

    while(1)
    {
       system("cls");
       puts("Choose an operation: ");
       puts("1\tDisplay contents of stack");
       puts("2\tPush data to stack");
       puts("3\tPop data from stack");
       puts("4\tExit");
       puts("\nEnter choice...");

       c = getch();
       fflush(stdin);

       switch(c)
       {
       case '1':
              system("cls");
```

```
                temp = top;
                printf("Stack contents:");
                while(temp != NULL)
                {
                    printf("\n%d", temp->data);
                    temp = temp->next;
                }

                printf("\n\nPress any key to return to menu...");
                getch();
                break;

        case '2':
                system("cls");

                printf("Enter the data to be pushed to stack: ");
                scanf("%d", &data);

                if(!push(data))
                    printf("\nData pushed successfully...Press any key to return to
                    menu...");
                else
                    printf("\nError: Stack overflow...Press any key to return to
                    menu...");
                getch();
                break;

        case '3':
                system("cls");

                printf("Popped value: %d \nPress any key to return to menu...",
                pop());
                getch();
                break;

        case '4':
                exit(0);

        default:
                system("cls");

                printf("Invalid input...Press any key to return to menu...");
                getch();
        }
    }
}
```