EXPERIMENT NO. 8

Ques 1 :- Write a program to implement Stack Data Structures. Implement the Push and Pop operation in the stack?

```
#include<stdio.h>
#include<stdlib.h>
struct stack
{
   int size;
   int top;
   int *s;
};
void push(struct stack *st){
   if(st->top == st->size-1)
      printf("Stack is Overflow\n");
   else{
      int x;
      printf("Enter element to be pushed into the stack: \n");
      scanf("%d",&x);
      st->top+=1;
      st->s[st->top] = x;

   }
```

```
}
int pop(struct stack *st){
   if(st->top == -1)
      printf("Stack is Underflow\n");
   else{
      int x = st->s[st->top];
      printf("Popping %d out of the stack\n", x);
      st->top-=1;
      return x;
   }
   return -1;
}
void Display(struct stack st)
{
   int i;
   for(i=st.top;i>=0;i--)
   {
      printf("%d ",st.s[i]);
   }
   printf("\n");
}
int main()
```

```
{
    struct stack st;
    st.top=-1;
    printf("Enter the size of the stack :\n");
    scanf("%d",&st.size);
    st.s = (int*)malloc(st.size*(sizeof(int)));
    printf("%d size of stack is created.\n",st.size);
    int choice;

    while(1){
        printf("\nChoose any of the following options:\n");
        printf(" 0: Exit          1: Push          2: Pop          3: Display\n");
        scanf("%d", &choice);

        switch(choice){
            case 0: exit(0);
            case 1: push(&st);
            break;
            case 2: pop(&st);
            break;
            case 3 : Display(st);
            break;
```

```
        default:
        printf("Please choose a correct option!");
    }
  }



}
```

Output of the Program :-

```
Enter the size of the stack :
5
5 size of stack is created.

Choose any of the following options:
 0: Exit            1: Push            2: Pop            3: Display
1
Enter element to be pushed into the stack:
10

Choose any of the following options:
 0: Exit            1: Push            2: Pop            3: Display
1
Enter element to be pushed into the stack:
20

Choose any of the following options:
 0: Exit            1: Push            2: Pop            3: Display
1
Enter element to be pushed into the stack:
30

Choose any of the following options:
 0: Exit            1: Push            2: Pop            3: Display
1
Enter element to be pushed into the stack:
40
```

```
Choose any of the following options:
 0: Exit            1: Push            2: Pop            3: Display
1
Enter element to be pushed into the stack:
50

Choose any of the following options:
 0: Exit            1: Push            2: Pop            3: Display
1
Stack is Overflow

Choose any of the following options:
 0: Exit            1: Push            2: Pop            3: Display
2
Popping 50 out of the stack

Choose any of the following options:
 0: Exit            1: Push            2: Pop            3: Display
2
Popping 40 out of the stack
```

Ques 2 :- Kindly use the stack data structure to reverse a string and if the string is palindrome or not?

```
#include<stdio.h>

#include<stdlib.h>

#include<string.h>


char* stack=NULL;

int size=50;

int top=-1;
```

```
void push(char x){
   if(stack==NULL)
   stack=(char*)malloc(size*sizeof(char));



   if(top==size-1)return;
    stack[++top]=x;

}

char pop(){
   if(top==-1)return '1';
   char x=stack[top--];
   return x;
}

void display(){
   for(int i=0;i<=top;i++)
   {
      printf("%c ",stack[i]);
   }
   printf("\n");
```

```
}

int main(){

    char name[20];
    printf("Enter the string :\n");
    gets(name);

    for(int i=0;i<strlen(name);i++)
    push(name[i]);

    char *store=(char*)malloc(strlen(name)*sizeof(char));
    int i=0;

    while(1){
        char x=pop();
        if(x=='1')
        break;
        store[i++]=x;
    }

    printf("\n");
```

```
    if(strcmp(store,name)==0)printf("palindrome");
    else printf("not palindrome");

    return 0;
}
```

Output of the Program :-

```
Enter the string :
malayalam

palindrome
PS E:\Data Structure and Algorithm In C\Experiment 8>
```

Ques 3 :- For the given string 2+3*5+8/2+6 convert into postfix and eventually solve this using stack?

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

```
struct Node
  {
    char data;
    struct Node *next;
  }*top=NULL;
void push(char x)
  {
    struct Node *t;
    t=(struct Node*)malloc(sizeof(struct Node));

    if(t==NULL)
    printf("stack is full\n");
    else
    {
      t->data=x;
      t->next=top;
      top=t;
    }

  }
char pop()
  {
```

```
    struct Node *t;
    char x=-1;

    if(top==NULL)
    printf("Stack is Empty\n");
    else
    {
      t=top;
      top=top->next;
      x=t->data;
      free(t);
    }
    return x;
  }
void Display()
  {
    struct Node *p;
    p=top;
    while(p!=NULL)
    {
      printf("%d ",p->data);
      p=p->next;
```

```
        }
        printf("\n");
    }
int isBalanced(char *exp)
    {
        int i;

        for(i=0;exp[i]!='\0';i++)
        {
        if(exp[i]=='(')
        {
            push(exp[i]);
        }
        else if(exp[i]==')')
        {
            if(top==NULL)
            return 0;
            pop();
            }
        }
        if(top==NULL)
        return 1;
```

```
    else
    return 0;
  }
int pre(char x)
  {
  if(x=='+' || x=='-')
  return 1;
  else if(x=='*' || x=='/')
  return 2;
  return 0;
  }

int isOperand(char x)
{
  if(x=='+' || x=='-' || x=='*' || x=='/')
  return 0;
  else
  return 1;

}

char * InToPost(char *infix)
```

```
{
    int i=0,j=0;
    char *postfix;
    int len=strlen(infix);
    postfix=(char *)malloc((len+2)*sizeof(char));

    while(infix[i]!='\0')
    {
        if(isOperand(infix[i]))
        {
            postfix[j++]=infix[i++];
        }
        else
        {
            if(pre(infix[i])>pre(top->data))
            {
                push(infix[i++]);
            }
            else
            {
                postfix[j++]=pop();
            }
```

```
        }
    }
    while(top!=NULL)
    postfix[j++]=pop();
    postfix[j]='\0';
    return postfix;
  }

int main()
{
 char *infix="2+3*5+8/2+6";
 push('#');

 char *postfix=InToPost(infix);

 printf("%s ",postfix);



 return 0;
}
```

Output of the Program :-

```
235*+82/+6+#
PS E:\Data Structure and Algorithm In C\Experiment 8>
```

Ques 4 :- Implement a functionality where a dedicated memory which was allocated using malloc() is about to get filled, reallocate the memory if the available memory is less than 10%?

```
#include<stdio.h>

#include<stdlib.h>

#include<stdbool.h>

#define intial_size 5

#define percent 10

int top = -1;

int stack[intial_size];

int recSize=intial_size;

int *data=NULL;


void push();

int pop();
```

```
bool memoryAlmostFull(int currentSize,int usedSize)
{
    float availablePercent = (currentSize - usedSize)/currentSize;
    return availablePercent<percent;
}
void push(int *data){
  if(top == recSize-1)
  {
  int usedSize = top+1;
  if(memoryAlmostFull(recSize,usedSize))
  {
    int newSize = recSize * 2;
    int *temp = (int *)realloc(data, newSize * sizeof(int));
    if(temp==NULL)
    {
      printf("Memory reallocation failed\n");
          return;
       } else {
          data = temp;
          recSize = newSize;
       }
  }
```

```
    }
        //printf("Overflow State: can't add more elements into the
stack\n");
    else{
        int x;
        printf("Enter element to be pushed into the stack: \n");
        scanf("%d", &x);
        top+=1;
        stack[top] = x;

    }
}

int pop(){
    if(top == -1)
        printf("Underflow State: Stack already empty, can't remove any
element\n");
    else{
        int x = stack[top];
        printf("Popping %d out of the stack\n", x);
        top-=1;
        return x;
    }
```

```
    return -1;
}
// void size()
// {
//     int usedSize = intial_size-top;
//     printf("Size left in Stack : %d",usedSize);
// }
int main()
{
    data = (int*)malloc(intial_size*sizeof(int));
    if(data==NULL)
    {
        printf("Memory allocation failed");
    }

    printf("STATIC ARRAY (Total Capacity: %d)\n",intial_size);
    int choice;

    while(1){
        printf("\nChoose any of the following options:\n");
        printf(" 0: Exit          1: Push          2: Pop          \n");
        scanf("%d", &choice);
```

```
        switch(choice){
            case 0: exit(0);
            case 1: push(data);
            break;
            case 2: pop();
            break;
            // case 3 : size();
            // break;
            default: printf("Please choose a correct option!");
        }
    }

    return 0;
}
```

Output of the Program :-

```
Choose any of the following options:
 0: Exit            1: Push            2: Pop
1
Enter element to be pushed into the stack:
10

Choose any of the following options:
 0: Exit            1: Push            2: Pop
1
Enter element to be pushed into the stack:
20

Choose any of the following options:
 0: Exit            1: Push            2: Pop
1
Enter element to be pushed into the stack:
30
```

```
Enter element to be pushed into the stack:
40

Choose any of the following options:
 0: Exit            1: Push            2: Pop
1
Enter element to be pushed into the stack:
50

Choose any of the following options:
 0: Exit            1: Push            2: Pop
1

Choose any of the following options:
 0: Exit            1: Push            2: Pop
60
Please choose a correct option!
Choose any of the following options:
 0: Exit            1: Push            2: Pop
1
Enter element to be pushed into the stack:
60
```