Assignment No. 9: Stack

A palindrome is a string of character that‘s the same forward and backward. Typically, punctuation, capitalization, and spaces are ignored. For example, “Poor Dan is in a droop” is a palindrome, as can be seen by examining the characters “poor danisina droop” and observing that they are the same forward and backward. One way to check for a palindrome is to reverse the characters in the string and then compare with them the original-in a palindrome, the sequence will be identical. Write C++ program with functions-

a) To print original string followed by reversed string using stack

b) To check whether given string is palindrome or not

Objectives:

1. To study stack using array
2. To check whether the given string is palindrome or not

Problem Statement:

A palindrome is a string of character that‘s the same forward and backward. Typically, punctuation, capitalization, and spaces are ignored. For example, ‖Poor Dan is in a droop‖ is a palindrome, as can be seen by examining the characters ―poor danisina droop‖ and observing that they are the same forward and backward. One way to check for a palindrome is to reverse the characters in the string and then compare with them the original-in a palindrome, the sequence will be identical. Write C++ program with functions-

1. To check whether given string is palindrome or not that uses a stack to determine whether a string is a palindrome.

2. To remove spaces and punctuation in string, convert all the Characters to lowercase, and then call above Palindrome checking function to check for a palindrome

3. To print string in reverse order using stack

Software Requirements:

Ubuntu Linux 14.04, GCC / G++ (editor)

Hardware Requirements:

Any processor above Pentium 4

Theory and Concept:

1. **Stack:**
    1. Stack is a LIFO (last in first out) structure. It is an ordered list of the same type of elements.
    2. A stack is a linear list where all insertions and deletions are permitted at only one end of list.
    3. When elements are added to stack it grows at one end. Similarly, when elements are deleted from stack, it shrinks at the same end.
2. **Stack Using Array:**
    1. Stack is a LIFO structure. Stack can be represented using array.
    2. A one dimensional array can be used to hold elements of stack.
    3. Another variable "top" is used to keep track of the index of the top most element.
    4. The following operations can be done on the stack by using array:

    a) **Initialization Of Stack:**

```
#include<iostrem>
#include<string.h>
#define MAX 100
using namespace std;

typedefstruct stack
{
        Char data[MAX];
         int top;
}stack;

voidinit(stack *stck)
{
          Char data[MAX];
          int top;
}stack;

voidinit(stack *stck)
{
          int I;
         for(i=0;i<MAX;i++)
         stck->data[i]='\0';
         stck->top=-1;
}
```

b) **Print Function:**

```
void print(stack stck)
          {
              int i;
              cout<<"\nStack elements are::";
              for(i=0;i<MAX;i++)
              cout<<stck.data[i];
              cout<<"\ttop="<<stck.top;
           }
```

c) **Isempty Condition:**

```
intisempty(stack stck)
      {
          returnstck.top==-1?1:0;
      }
```

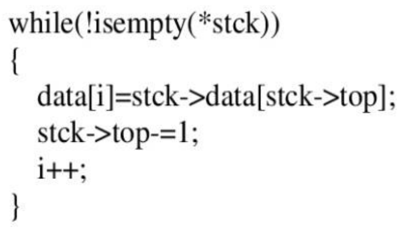
d) **Isfull Condition:**

```
intisfull(stack stck)
      {
          returnstck.top==MAX-1?1:0;
      }
```

e) **Push Function:**

```
for(i=0;data[i]!='\0';i++)
       {
          stck->top+=1;
           stck->data[stck->top]=data[i];
       }
```

f) **Pop Function:**



```
while(!isempty(*stck))
{
   data[i]=stck->data[stck->top];
   stck->top-=1;
   i++;
}
```

3. Palindrome:

A palindrome is a word, sentence, or number that reads the same from left to right as from right to left. In this program, we take two cases to check whether the string is palindrome or not:

a) Keep the string as it is:

As given string “Poor Dan is in a droop” is reversed as it is, it becomes “poord a nisina DrooP” which is not similar to given string. Therefore, given string is not palindrome.

b) Remove all spaces, punctuations and convert uppercases into lowercases:

Given string is “Poor Dan is in a droop”. If we convert all uppercases into lower cases it becomes “poor dan is in a droop”. Now remove all punctuations and spaces it becomes “poor danisina droop”. If we reversed this string it becomes “poor danisina dreoop” which is similar to that. Therefore, given string is palindrome.

```
init(&stck);
for(i=0;data[i]!=’\0’;i++)
   {
 if(data[i]!=’ ‘)
        {
      if(data[i]>=65 && data[i]<=90)
               data1[j]=data[i]+32;
     else
              data1[j]=data[i];
              j++;
        }
   }
```

Algorithm:

Step 1: Start

Step 2: Declare characters data[MAX] and data1[MAX].

Step 3: Declare integers top, i, j, ch

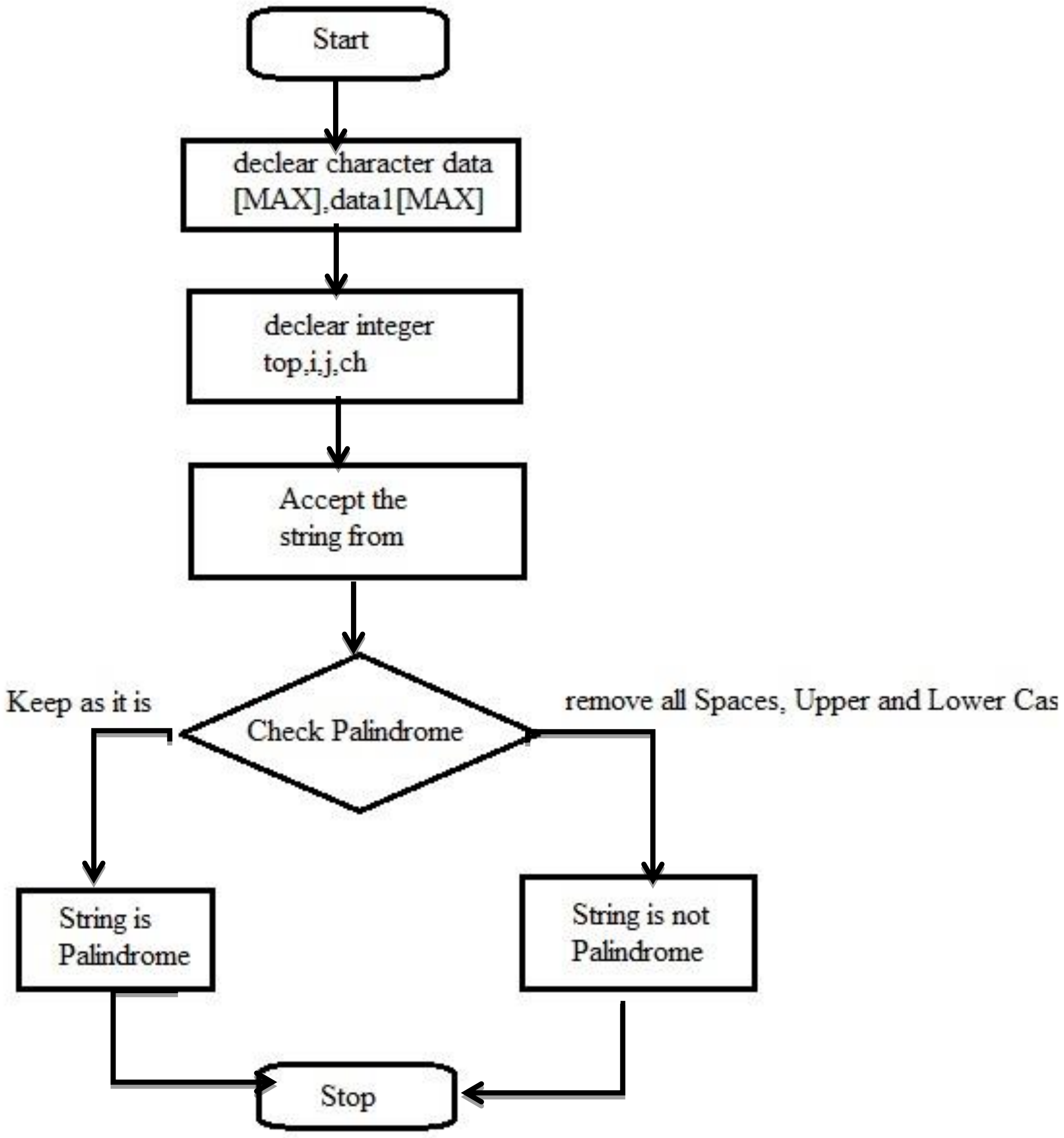
Step 4: Accept the string from user

Step 5: Check given string is palindrome or not and display it. If it is palindrome then goto step 8 else goto step 6.

Step 6: Change upper cases to lower cases and remove all spaces

Step 7: Again check the string is palindrome or not and display it

Step 8: Stop



Conclusion: With the help of stack using array, we successfully check whether the string is palindrome or not.

```
#include <iostream>
#include<stdio.h>
#include <stdlib.h>
#include <string.h>
using namespace std;


class stack
{
    char s[25];
    int top;

public:
    stack()
        {
            top=-1;
        }
    void push(char val);
    char pop();
    int isfull();
    int isempty();

};

class str
{
    char inputstr[25], revstr[25];
    stack stobj;

public:
    void readstring();
    void revstring();
    void convertstring();
    void checkpalindrome();

};

int stack::isempty()
{
    if(top==-1)
        return 1;
    else
        return 0;
}

int stack::isfull()
{
    if(top==24)
        return 1;
    else
        return 0;
```

```
}

void stack::push(char val)
{
   if(!isfull())
   {
      top++;
      s[top]=val;
   }
   else
      cout<<endl<<"stack overflows/full......"<<endl;
}

char stack::pop()
{
   char val='\0';
   if(!isempty())
   {
      val=s[top];
      top--;

   }
   else
      cout<<endl<<"stack underflow/empty......"<<endl;
   return val;
}




void str::readstring()
{
   cout<<"\nEnter string  ";
   gets(inputstr);
   cout<<"\nYou entered:->  "<<inputstr;
}

void str::revstring()
{
   int i;
   char ch;
   for(i=0;inputstr[i]!='\0';i++)
      stobj.push(inputstr[i]);

   i=0;
   cout<<"\n\nAfter reverse your string:-> ";
   while(!stobj.isempty())
   {
      ch=stobj.pop();
      cout<<ch;
      revstr[i]=ch;
```

```
        i++;
    }
}

void str::convertstring()
{
    int i,j=0;
    char tempstr[25];
    for(i=0;inputstr[i]!='\0';i++)
    {
        if(inputstr[i]>=97  && inputstr[i]<=122)
        {
            tempstr[j]=inputstr[i];
            j++;
        }
        else if(inputstr[i]>=65 && inputstr[i]<=190)
        {
            tempstr[j]=inputstr[i]+32;
            j++;
        }

    }
    tempstr[j]='\0';
    strcpy(inputstr,tempstr);
    cout<<"\n\nYour converted string:-> "<<inputstr;
}


void str::checkpalindrome()
{
    cout<<"\n\n\n\n\n";
    cout<<"\nCheck for palindrome"<<endl;

    readstring();
    convertstring();
    revstring();

    if(strcmp(inputstr,revstr)==0)
        cout<<"\n\nYour string is PALINDROME";
    else
        cout<<"\n\nYour string is NOT PALINDROME";

}


int main()
{
    str obj;
    obj.readstring();
    obj.convertstring();
    obj.revstring();
    obj.checkpalindrome();
    return 0;
}



/*=============================================================================
============OUTPUT==================================================
=============================================

Enter string   N 123iti124164 n

You entered:->   N 123iti124164 n

Your converted string:-> nitin

After reverse your string:-> nitin



Check for palindrome

Enter string  N 123iTI165265 n

You entered:->  N 123iTI165265 n

Your converted string:-> nitin

After reverse your string:-> nitin

Your string is PALINDROME

 */
```