# 15B17CI371 – Data Structures Lab

# ODD 2024

# Week 2-LAB A

# Practice Lab

**1.What is output of the following segment of code? Justify your output by**

**performing a dry run i.e.,showcasing each step and change in value for each**

**variable.**

**stackType<int> stack;**

**int x,y;**

**x=4;**

**y=0;**

**stack.push(7);**

**stack.push(x);**

**stack.push(x + 5);**

**y=stack.top();**

**stack.pop();**

**stack.push(x + y);**

**stack.push(y-2);**

**stack.push(3);**

**x=stack.top();**

**stack.pop();**

**cout<<"x="<<x<<endl;**

**cout<<"y="<<y<<endl;**

**while(!stack.isEmptyStack())**

**{**

**cout<<stack.top()<<endl;**

**stack.pop();**

**}**

**Justification :**

x**=**4

y=0

**Operations on Stack:**

1. **stack.push(7);**
   * The stack now contains: [7]
2. **stack.push(x);**
   * x is 4.
   * The stack now contains: [7,4]
3. **stack.push(x + 5);**
   * x + 5 is 4 + 5=9.
   * The stack now contains: [7,4,9]
4. **y=stack.top();**
   * stack.top() returns the top element,which is 9.
   * y is now 9.
5. **stack.pop();**
   * The top element(9) is removed from the stack.
   * The stack now contains: [7,4]
6. **stack.push(x + y);**
   * x + y is 4 + 9=13.
   * The stack now contains: [7,4,13]
7. **stack.push(y-2);**
   * y-2 is 9-2=7.
   * The stack now contains: [7,4,13,7]
8. **stack.push(3);**
   * The stack now contains: [7,4,13,7,3]
9. **x=stack.top();**
   * stack.top() returns the top element,which is 3.
   * x is now 3.
10. **stack.pop();**
    * The top element(3) is removed from the stack.
    * The stack now contains: [7,4,13,7]
11. The loop prints and removes each element from the stack until it is empty.
12. The stack contains [7,4,13,7] before the loop starts.

**Output :**

**x=3**

**y=9**

**7**

**13**

**4**

**7**

**2. Write a program that uses a stack to print the prime factors of a positive integer**

**in descending order.**

#include <iostream>

using namespace std;

struct Node

{

int data;

Node\* next;

};

class CustomStack

{

Node\* top;

public:

CustomStack()

{

top=NULL;

}

bool isEmpty()

{

return top==NULL;

}

void push(int value)

{

Node\* newNode=new Node;

newNode->data=value;

newNode->next=top;

top=newNode;

}

int pop()

{

if(isEmpty())

{

cout<<"Stack is empty."<<endl;

return -1;

}

Node\* temp=top;

int poppedValue=temp->data;

top=top->next;

delete temp;

return poppedValue;

}

int peek()

{

if(isEmpty())

{

cout<<"Stack is empty."<<endl;

return -1;

}

return top->data;

}

};

bool is\_prime(int n)

{

if(n<=1)

return false;

for(int i=2;i\*i<=n;i++)

if(n%i==0)

return false;

return true;

}

void prime\_factors(int n)

{

CustomStack stack;

for(int i=2;i<=n;i++)

while(is\_prime(i)&&n%i==0)

{

stack.push(i);

n/=i;

}

cout<<"Prime factors in descending order: ";

while(!stack.isEmpty())

cout<<stack.pop()<<" ";

cout<<endl;

}

int main()

{

int n;

cout<<"Enter a positive integer: ";

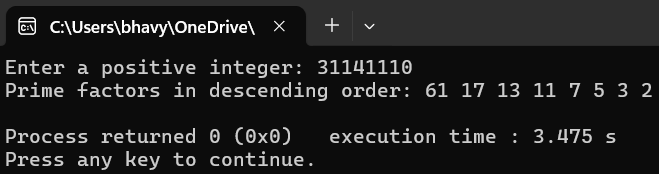
cin>>n;

prime\_factors(n);

return 0;

}

**Output :**



**3. Write a program to split a stack into two stacks with one containing the bottom**

**half elements and the second the remaining elements;and to combine two**

**stacks into one by placing all elements of the second stack on top of those of**

**the first stack.**

#include <iostream>

using namespace std;

#define MAX\_SIZE 100

class Stack

{

int arr[MAX\_SIZE];

int top;

public:

Stack():top(-1){}

bool isEmpty()

{

return top==-1;

}

bool isFull()

{

return top==MAX\_SIZE-1;

}

void push(int value)

{

if(isFull())

{

cout<<"Stack is full!"<<endl;

return;

}

arr[++top]=value;

}

int pop()

{

if(isEmpty())

{

cout<<"Stack is empty!"<<endl;

return -1;

}

return arr[top--];

}

int peek()

{

if(isEmpty())

{

cout<<"Stack is empty!"<<endl;

return -1;

}

return arr[top];

}

void print()

{

for(int i=top;i>=0;--i)

cout<<arr[i]<<" ";

cout<<endl;

}

};

void splitStack(Stack &original,Stack &bottomHalf,Stack &topHalf)

{

int temp[MAX\_SIZE];

int size=0;

while(!original.isEmpty())

temp[size++]=original.pop();

int mid=size/2;

for(int i=mid-1;i>=0;--i)

bottomHalf.push(temp[i]);

for(int i=size-1;i>=mid;--i)

topHalf.push(temp[i]);

}

Stack combineStacks(Stack &first,Stack &second)

{

Stack tempStack1,tempStack2,tempStack;

while(!first.isEmpty())

tempStack1.push(first.pop());

while(!second.isEmpty())

tempStack2.push(second.pop());

while(!tempStack1.isEmpty())

tempStack.push(tempStack1.pop());

while(!tempStack2.isEmpty())

tempStack.push(tempStack2.pop());

return tempStack;

}

int main()

{

int n,a;

Stack originalStack;

Stack bottomHalfStack;

Stack topHalfStack;

cout<<"Input the number of elements : ";

cin>>n;

cout<<"Input the elements : ";

for(int i=0;i<n;i++)

{

cin>>a;

originalStack.push(a);

}

cout<<"Original Stack: ";

originalStack.print();

splitStack(originalStack,bottomHalfStack,topHalfStack);

cout<<"Bottom Half Stack: ";

bottomHalfStack.print();

cout<<"Top Half Stack: ";

topHalfStack.print();

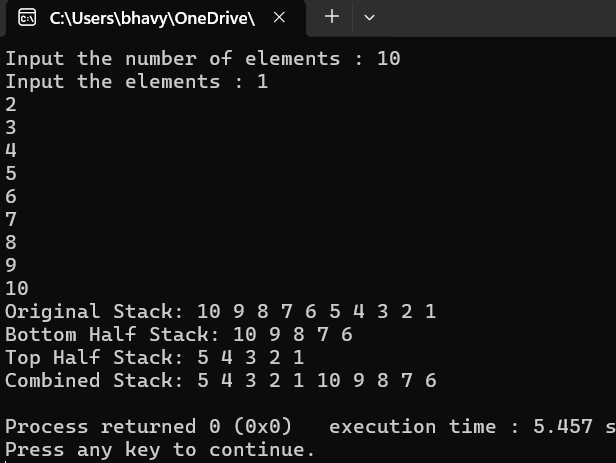
Stack recombine=combineStacks(bottomHalfStack,topHalfStack);

cout<<"Combined Stack: ";

recombine.print();

}

**Output :**

****

**4. Write a program to convert a number from decimal notation to a number**

**expressed in a number system whose base(or radix) is a number between 2**

**and 9. The conversion is performed by repetitious division by the base to which**

**a number is being converted and then taking the remainders of division in the**

**reverse order.**

**For example,in converting to binary,number 6 requires three such divisions:**

**6/2=3 remainder 0,**

**3/2=1 remainder 1,and finally,**

**1/2=0 remainder 1.**

**The remainders 0,1,and 1 are put in the reverse order so that binary**

**equivalent of 6 is equal to 110.**

**Use stacks/queues to implement the conversion of integer of base 10 to binary**

#include <iostream>

#include <stack>

using namespace std;

void convertToBase(int decimalNumber,int base)

{

if(base<2||base>9)

{

cout<<"Base must be between 2 and 9."<<endl;

return;

}

stack<int> remainderStack;

if(decimalNumber==0)

{

cout<<"0"<<endl;

return;

}

while(decimalNumber>0)

{

int remainder=decimalNumber%base;

remainderStack.push(remainder);

decimalNumber/=base;

}

while(!remainderStack.empty())

{

cout<<remainderStack.top();

remainderStack.pop();

}

cout<<endl;

}

int main()

{

int decimalNumber,base;

cout<<"Enter a decimal number : ";

cin>>decimalNumber;

cout<<"Enter the base(between 2 and 9) : ";

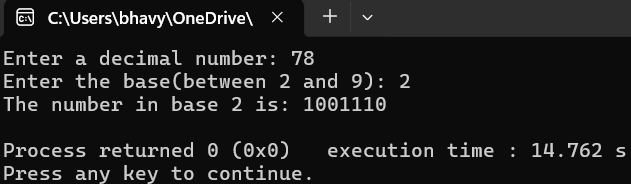
cin>>base;

cout<<"The number in base "<<base<<" is : ";

convertToBase(decimalNumber,base);

}

**Output :**



**6. Write a program to check for balancing symbols(parentheses forms) in the**

**following languages:(),[],{}**

#include <iostream>

#include <stack>

using namespace std;

bool isOpeningSymbol(char c)

{

return c=='('||c=='['||c=='{';

}

bool isClosingSymbol(char c)

{

return c==')'||c==']'||c=='}';

}

bool isMatchingPair(char open,char close)

{

return(open=='('&&close==')') ||(open=='['&&close==']') ||(open=='{'&&close=='}');

}

bool areSymbolsBalanced(string expr)

{

stack<char> st;

for(char c:expr)

{

if(isOpeningSymbol(c))

st.push(c);

else if(isClosingSymbol(c))

{

if(st.empty()||!isMatchingPair(st.top(),c))

return false;

st.pop();

}

}

return st.empty();

}

int main()

{

string expr;

cout<<"Enter a string containing symbols((),[],{}): ";

cin>>expr;

if(areSymbolsBalanced(expr))

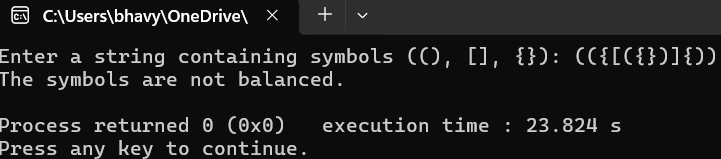
cout<<"The symbols are balanced."<<endl;

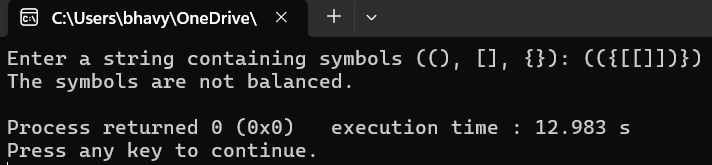
else

cout<<"The symbols are not balanced."<<endl;

}

**Output :**

****

****

**7. Write a program to compress a given text by removing whitespaces and**

**replacing continuously repeated character by character followed by no. of time,**

**it is repeated.Use queue data structure.**

#include <iostream>

#include <queue>

using namespace std;

string compressText(string text)

{

queue<char> q;

string compressed="";

for(char c:text)

if(!isspace(c))

q.push(c);

while(!q.empty())

{

char currentChar=q.front();

q.pop();

int count=1;

while(!q.empty()&&q.front()==currentChar)

{

count++;

q.pop();

}

compressed+=currentChar;

if(count>1)

compressed+=to\_string(count);

}

return compressed;

}

int main()

{

string text;

cout<<"Enter the text to compress: ";

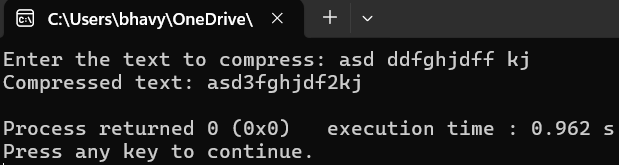
getline(cin,text);

string compressedText=compressText(text);

cout<<"Compressed text: "<<compressedText<<endl;

}

**Output :**

****

**8. Write the definition of the function ‘moveNthFront’ that takes as a parameter a**

**positive integer,n. The function moves the nth element of the queue to the**

**front. The order of the remaining elements remains unchanged.**

**example**

**Input queue={5,11,34,67,43,55} and n=3.**

**After a call to the function moveNthFront,**

**Output queue={34,5,11,67,43,55}.**

#include <queue>

#include <iostream>

using namespace std;

void moveNthFront(queue<int>& q,int n)

{

int size=q.size();

queue<int> temp1,temp2;

for(int i=1;i<n;i++)

{

temp1.push(q.front());

q.pop();

}

int nthElement=q.front();

q.pop();

temp2.push(nthElement);

while(!temp1.empty())

{

temp2.push(temp1.front());

temp1.pop();

}

while(!q.empty())

{

temp2.push(q.front());

q.pop();

}

q=temp2;

}

int main()

{

int n,a;

queue<int> q;

cout<<"Input the number of elements : “;;

cin>>n;

for(int i=0;i<n;i++)

{

cin>>a;

q.push(a);

}

cout<<"Input the value of n : ";

cin>>n;

if(n<=0||n>q.size())

{

cout<<"Invalid value of n. It must be between 1 and the size of the queue.";

return 0;

}

moveNthFront(q,n);

cout<<"Queue after moving the "<<n<<"rd element to the front: ";

while(!q.empty())

{

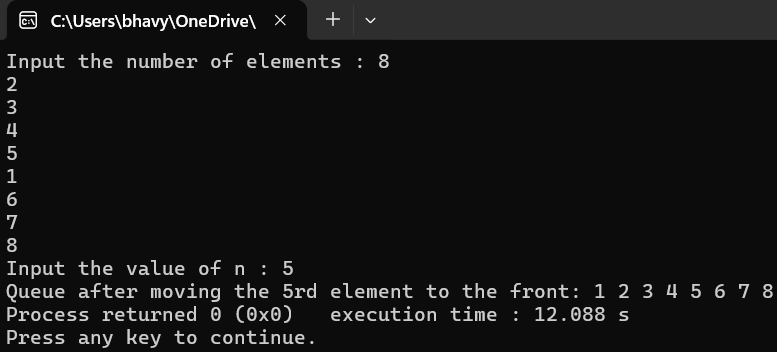
cout<<q.front()<<" ";

q.pop();

}

}

**Output :**

****

**9. Write a program that reads a line of text,changes each uppercase letter to**

**lowercase,and places each letter both in a queue and onto a stack. The**

**program should then verify whether the line of text is a palindrome.**

#include <iostream>

#include <queue>

#include <stack>

using namespace std;

bool isPalindrome(const string& text)

{

queue<char> q;

stack<char> s;

for(char ch:text)

{

if(ch>='A'&&ch<='Z')

ch=tolower(ch);

q.push(ch);

s.push(ch);

}

while(!q.empty())

{

if(q.front()!=s.top())

return false;

q.pop();

s.pop();

}

return true;

}

int main()

{

string text;

cout<<"Enter a line of text : ";

getline(cin,text);

if(isPalindrome(text))

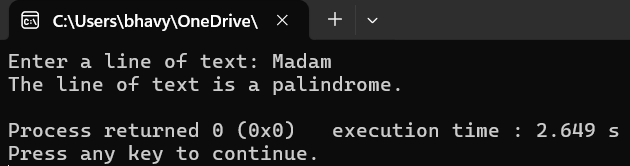
cout<<"The line of text is a palindrome."<<endl;

else

cout<<"The line of text is not a palindrome."<<endl;

}

**Output :**

****

**10. A string of characters is given.**

**A scientist is interested in a very typical pattern. He wishes to reverse all the**

**characters which lies inside 2 substrings namely S1,and S2. S1 is the string of**

**any length but starts from X and ends with Y. S2,starts from Y and ends with**

**X.**

**Example:**

**Input: “ABXNNYPEROYABCDCXT**

**Output: OREP.**

**Write a program to solve this problem**

#include <iostream>

#include <cstring>

using namespace std;

string reverseBetweenSubstrings(string& input)

{

size\_t start=input.find("X");

if(start==string::npos||input.find("Y",start)==string::npos)

return "Invalid input: S1 not found.";

start=input.find("Y",start) + 1;

size\_t end=input.find("Y",start);

if(end==string::npos||input.find("X",end)==string::npos)

return "Invalid input: S2 not found.";

end=input.find("Y",end);

if(start>=end)

return "Invalid input: S1 and S2 overlap.";

string between=input.substr(start,end-start);

string s;

int i=0 ;

while(between[i])

{

s=s+between[i];

i++;

}

return s;

}

int main()

{

string text;

cout<<"Input a string : ";

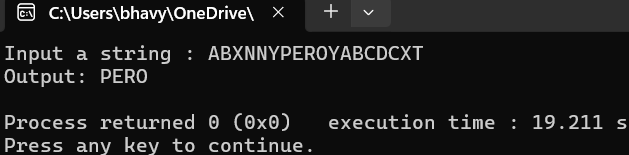
getline(cin,text);

string result=reverseBetweenSubstrings(text);

cout<<"Output: "<<result<<endl;

}

**Output :**

****