**LAB:-9(STACK)**

**Q :- Write a menu-driven program to implement stack using array**

*//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*This program is written by Manan Jain(211B173)\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//*

#include<iostream>

using namespace std;

class STACK

{

public:

int top;

int arr[5];

STACK()

{top=-1; }

bool isEmpty()

{

if(top<4)

return true;

else

return false;}

void push(int val)

{

if(isEmpty())

{

arr[top+1]=val;

top=top+1;

}

else

cout<<"Stack Overflow!"<<endl;

}

void pop()

{

int p=0;

p=arr[top];

if(top==-1)

cout<<"Stack Underflow!"<<endl;

else

{

top=top-1;

cout<<"Popped Value : "<<p<<endl;

}

}

void display()

{

cout<<"STACK : "<<endl;

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

{

cout<<arr[i]<<endl;

}

cout<<endl;

}

};

int main()

{

STACK s1;

int choice;

int data;

while(1)

{

cout<<"\*\*\*Stack Menu\*\*"<<endl;

cout<<"1.push()"<<endl;

cout<<"2.pop()"<<endl;

cout<<"3.display()"<<endl;

cout<<"4.Exit"<<endl;

cout<<"Enter Choice(1-4):";

cin>>choice;

switch(choice)

{

case 1:

cout<<"Enter Element to push : ";

cin>>data;

s1.push(data);

cout<<endl;

break;

case 2:

s1.pop();

break;

case 3:

s1.display();

break;

case 4:

break;

default:

cout<<"Enter Valid Choice"<<endl;

break;

}

if(choice==4)

break;

}

return 0;

}

**Q :- Write a menu-driven program to using array implement Linked List**

*//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*This program is written by Manan Jain(211B173)\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//*

#include<iostream>

using namespace std;

class Node

{

public:

Node\* NEXT;

int info;

Node()

{

NEXT=NULL;

info=0;

}

Node(int val)

{

NEXT=NULL;

info=val;

}

};

class STACK

{

public:

Node\* start;

STACK()

{

start=NULL;

}

Node\* CreateS()

{

return start;

}

void push(Node\*\* top,int data)

{

Node \*new\_node= new Node(data);

if(new\_node==NULL)

{

cout<<"STACK OVERFLOW"<<endl;

}

else

{

if(\*top==NULL)

{

\*top=new\_node;

new\_node->NEXT=NULL;

}

else

{

new\_node->NEXT=\*top;

\*top=new\_node;

}

}

}

void pop(Node\*\* top)

{

Node\* temp;

temp=\*top;

if(temp==NULL)

{

cout<<"STACK UNDERFLOW"<<endl;

}

else

{

int p;

p=temp->info;

cout<<"Popped Value : "<<p<<endl;

\*top=temp->NEXT;

delete temp;

}

}

void display(Node\*\* top)

{

Node\* temp=\*top;

cout<<"STACK : "<<endl;

while(temp!=NULL)

{

cout<<temp->info<<endl;

temp=temp->NEXT;

}

cout<<endl;

}

};

int main()

{

STACK \*s1 = new STACK();

Node\* top=s1->CreateS();

int choice;

int data;

while(1)

{

cout<<"\*\*\*Stack Menu\*\*\*"<<endl;

cout<<"1.push()"<<endl;

cout<<"2.pop()"<<endl;

cout<<"3.display()"<<endl;

cout<<"4.Exit"<<endl;

cout<<"Enter Choice(1-4):";

cin>>choice;

cout<<endl;

switch(choice)

{

case 1:

cout<<"Enter Element to push : ";

cin>>data;

s1->push(&top,data);

cout<<endl;

break;

case 2:

s1->pop(&top);

break;

case 3:

s1->display(&top);

break;

case 4:

break;

default:

cout<<"Enter Valid Choice"<<endl;

break;

}

if(choice==4)

break;

}

return 0;

}

**Q :- WAP to convert an expression from postfix to infix.**

*//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*This program is written by Manan Jain(211B173)\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//*

#include <bits/stdc++.h>

using namespace std;

bool isOperand(char x) {

return (x >= 'a' && x <= 'z') || (x >= 'A' && x <= 'Z');

}

string infixConversion(string postfix) {

stack<string> infix;

for (int i=0; postfix[i]!='\0'; i++) {

if (isOperand(postfix[i])) {

string op(1, postfix[i]);

infix.push(op);

} else {

string op1 = infix.top();

infix.pop();

string op2 = infix.top();

infix.pop();

infix.push("{"+op2+postfix[i]+op1 +"}");

}

}

return infix.top();

}

int main() {

string postfix = "xyae+/%";

cout<<"The infix conversion of the postfix expression '"<<postfix<<"' is : ";

cout<<infixConversion(postfix);

return 0;

}

**Q :- WAP to convert an expression from infix to postfix.**

*//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*This program is written by Manan Jain(211B173)\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//*

#include<iostream>

#include<stack>

#include<locale>

using namespace std;

int preced(char ch) {

   if(ch == '+' || ch == '-') {

      return 1;

   }else if(ch == '\*' || ch == '/') {

      return 2;

   }else if(ch == '^') {

      return 3;

   }else {

      return 0;

   }

}

string inToPost(string infix ) {

   stack<char> stk;

   stk.push('#');

   string postfix = "";

   string::iterator it;

   for(it = infix.begin(); it!=infix.end(); it++) {

      if(isalnum(char(\*it)))

         postfix += \*it;

      else if(\*it == '(')

         stk.push('(');

      else if(\*it == '^')

         stk.push('^');

      else if(\*it == ')') {

         while(stk.top() != '#' && stk.top() != '(') {

            postfix += stk.top();

            stk.pop();

         }

         stk.pop();

      }else {

         if(preced(\*it) > preced(stk.top()))

            stk.push(\*it); //push if precedence is high

         else {

            while(stk.top() != '#' && preced(\*it) <= preced(stk.top())) {

               postfix += stk.top();

               stk.pop();

            }

            stk.push(\*it);

         }  }}

   while(stk.top() != '#') {

      postfix += stk.top();

      stk.pop();}

   return postfix;

}

int main() {

   string infix = "x^y/(5\*z)+2";

   cout << "Postfix Form Is: " << inToPost(infix) << endl;

}

**Q :- WAP to convert an expression from infix to prefix**

*//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*This program is written by Manan Jain(211B173)\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//*

#include<iostream>

#include<stack>

#include<locale> //for function isalnum()

#include<algorithm>

using namespace std;

int preced(char ch) {

   if(ch == '+' || ch == '-') {

      return 1;    //Precedence of + or - is 1

   }else if(ch == '\*' || ch == '/') {

      return 2;    //Precedence of \* or / is 2

   }else if(ch == '^') {

      return 3;    //Precedence of ^ is 3

   }else {

      return 0;

   }

}

string inToPost(string infix) {

   stack<char> stk;

   stk.push('#');

   string postfix = "";

   string::iterator it;

   for(it = infix.begin(); it!=infix.end(); it++) {

      if(isalnum(char(\*it)))

         postfix += \*it;

      else if(\*it == '(')

         stk.push('(');

      else if(\*it == '^')

         stk.push('^');

      else if(\*it == ')') {

         while(stk.top() != '#' && stk.top() != '(') {

            postfix += stk.top();

            stk.pop();

         }

         stk.pop();

      }else {

         if(preced(\*it) > preced(stk.top()))

            stk.push(\*it);

         else {

            while(stk.top() != '#' && preced(\*it) <= preced(stk.top())) {

               postfix += stk.top();

               stk.pop();

            }

            stk.push(\*it);

         }

      }

   }

   while(stk.top() != '#') {

      postfix += stk.top();

      stk.pop();

   }

   return postfix;

}

string inToPre(string infix) {

   string prefix;

   reverse(infix.begin(), infix.end());

   string::iterator it;

   for(it = infix.begin(); it != infix.end(); it++) {

      if(\*it == '(')

         \*it = ')';

      else if(\*it == ')')

         \*it = '(';

   }

   prefix = inToPost(infix);

   reverse(prefix.begin(), prefix.end());

   return prefix;

}

int main() {

   string infix = "x^y/(5\*z)+2";

   cout << "Prefix Form Is: " << inToPre(infix) << endl;}

**Q :- WAP to evaluate postfix expression**

*//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*This program is written by Manan Jain(211B173)\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//*

#include<iostream>

#include<cmath>

#include<stack>

using namespace std;

float scanNum(char ch) {

   int value;

   value = ch;

   return float(value-'0');   }

int isOperator(char ch) {

   if(ch == '+'|| ch == '-'|| ch == '\*'|| ch == '/' || ch == '^')

      return 1;

   return -1;   }

int isOperand(char ch) {

   if(ch >= '0' && ch <= '9')

      return 1;

   return -1;  }

float operation(int a, int b, char op) {   if(op == '+')

      return b+a;

   else if(op == '-')

      return b-a;

   else if(op == '\*')

      return b\*a;

   else if(op == '/')

      return b/a;

   else if(op == '^')

      return pow(b,a);

   elsereturn INT\_MIN;  }

float postfixEval(string postfix) {

   int a, b;

   stack<float> stk;

   string::iterator it;

   for(it=postfix.begin(); it!=postfix.end(); it++) {

      if(isOperator(\*it) != -1) {

         a = stk.top();

         stk.pop();

         b = stk.top();

         stk.pop();

         stk.push(operation(a, b, \*it));

      }else if(isOperand(\*it) > 0) {

         stk.push(scanNum(\*it)); } }

   return stk.top();}

int main() {

   string post = "53+62/\*35\*+";

   cout << "The result is: "<<postfixEval(post);}

**Q :- WAP to implement tower of Hanoi puzzle**

*//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*This program is written by Manan Jain(211B173)\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//*

#include<iostream>

using namespace std;

void TOH(int d, char t1, char t2, char t3)

{

if(d==1)

{cout<<"\nShift top disk from tower "<<t1<<" to tower "<<t2;

return;}

TOH(d-1,t1,t3,t2);

cout<<"\nShift top disk from tower "<<t1<<" to tower "<<t2;

TOH(d-1,t3,t2,t1);}

int main()

{int disk;

cout<<"Enter the number of disks: "; cin>>disk;

if(disk<1)

cout<<"There are no disks to shift";

else

cout<<"There are "<<disk<<" disks in tower 1\n";

TOH(disk, '1','2','3');

cout<<"\n\n"<<disk<<" disks in tower 1 are shifted to tower 2";

return 0;}

**LAB:-10(Queue)**

**Q :- Write a menu driven program to implement linear queue using array**

*//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*This program is written by Manan Jain(211B173)\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//*

#include<iostream>

using namespace std;

class qarr

{

public:

int arr[5];

int frnt;

int rear;

qarr()

{

frnt=-1;

rear=-1;

}

void Enqueue(int value)

{

if(frnt==-1 && rear ==-1)

{

frnt=0;

rear=0;

}

else if (rear==4)

{

if(frnt==0)

cout<<"QUEUE IS FULL!";

int temp=frnt;

int i=0;

while(temp<=rear)

{

arr[i]=arr[temp];

temp+=1;

i+=1;

}

rear=rear-frnt+1;

frnt=0;

}

else

rear+=1;

arr[rear]=value;

}

void Dequeue()

{

int item=arr[frnt];

if(rear==-1 && frnt==-1)

{

cout<<"EMPTY QUEUE"<<endl;

}

else if(frnt==rear)

{

frnt=-1;

rear=-1;

cout<<"DELETED VALUE:"<<item;

}

else

{

frnt++;

cout<<"DELETED VALUE:"<<item;

}

}

void disfrnt()

{

if(frnt==-1 && rear==-1)

{

cout<<"EMPTY QUEUE"<<endl;

}

else

cout<<"Front : "<<arr[frnt];

}

void display()

{

if(frnt==-1 && rear==-1)

{

cout<<"EMPTY QUEUE"<<endl;

}

else{

cout<<"Queue : ";

for(int i=frnt;i<=rear;i++)

{

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

}

}

}

};

int main()

{

qarr q1;

int choice;

int data;

while(1)

{

cout<<endl<<"\*\*\*Queue Menu\*\*"<<endl;

cout<<"1.enqueue()"<<endl;

cout<<"2.dequeue()"<<endl;

cout<<"3.displayfront()"<<endl;

cout<<"4.displayall()"<<endl;

cout<<"5.exit()"<<endl;

cout<<"Enter Choice(1-5):";

cin>>choice;

switch(choice)

{

case 1:

cout<<"Enter Element to enqueue : ";

cin>>data;

q1.Enqueue(data);

cout<<endl;

break;

case 2:

q1.Dequeue();

break;

case 3:

q1.disfrnt();

break;

case 4:

q1.display();

break;

case 5:

break;

default:

cout<<"Enter Valid Choice"<<endl;

break;

}

if(choice==5)

break;

}

return 0;

}

**Q :- Write a menu driven program to implement circular queue using array**

*//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*This program is written by Manan Jain(211B173)\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//*

#include<iostream>

using namespace std;

class qarr

{

public:

int arr[5];

int frnt;

int rear;

qarr()

{

frnt=-1;

rear=-1;

}

void Enqueue(int value)

{

if(frnt==-1 && rear ==-1)

{

frnt=0;

rear=0;

}

else if (rear==4)

{

if(frnt==0)

{

cout<<"QUEUE IS FULL!"<<endl;

return;

}

else

rear=0;

}

else if((rear+1)==(frnt))

{

cout<<"QUEUE IS FULL!"<<endl;

return;

}

else

rear+=1;

arr[rear]=value;

}

void Dequeue()

{

if(rear==-1 || frnt==-1)

{

cout<<"EMPTY QUEUE"<<endl;

return;

}

int item=arr[frnt];

if(frnt==rear)

{

frnt=-1;

rear=-1;

cout<<"DELETED VALUE:"<<item;

}

else if(frnt==4)

{

frnt=0;

cout<<"DELETED VALUE:"<<item;

}

else

{

frnt=frnt+1;

cout<<"DELETED VALUE:"<<item;

}

}

void disfrnt()

{

if(frnt==-1 && rear==-1)

{

cout<<"EMPTY QUEUE"<<endl;

}

else

cout<<"Front : "<<arr[frnt];

}

void display()

{

if(frnt==-1 && rear==-1)

{

cout<<"EMPTY QUEUE"<<endl;

}

else if(frnt<=rear)

{

for(int i=frnt;i<=rear;i++)

{

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

}

}

else

{

for(int i=frnt;i<5;i++)

{

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

}

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

{

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

}

}

}

};

int main()

{

qarr q1;

int choice;

int data;

while(1)

{

cout<<endl<<"\*\*\*Queue Menu\*\*"<<endl;

cout<<"1.enqueue()"<<endl;

cout<<"2.dequeue()"<<endl;

cout<<"3.displayfront()"<<endl;

cout<<"4.displayall()"<<endl;

cout<<"5.exit()"<<endl;

cout<<"Enter Choice(1-5):";

cin>>choice;

switch(choice)

{

case 1:

cout<<"Enter Element to enqueue : ";

cin>>data;

q1.Enqueue(data);

cout<<endl;

break;

case 2:

q1.Dequeue();

break;

case 3:

q1.disfrnt();

break;

case 4:

q1.display();

break;

case 5:

break;

default:

cout<<"Enter Valid Choice"<<endl;

break;

}

if(choice==5)

break;

}

return 0;

}

**Q :- Write a menu driven program to implement linear queue using linked list**

*//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*This program is written by Manan Jain(211B173)\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//*

#include<iostream>

using namespace std;

class Node

{

public:

Node\* NEXT;

int info;

Node()

{

NEXT=NULL;

info=0;

}

Node(int val)

{

NEXT=NULL;

info=val;

}

};

class QUEUE

{

public:

Node\* start;

Node\* tail;

QUEUE()

{

start=NULL;

tail=NULL;

}

Node\* sCreateQ()

{

return start;

}

Node\* tCreateQ()

{

return tail;

}

void Enqueue(Node\*\* top,Node\*\* Tail,int data)

{

Node \*new\_node= new Node(data);

if(new\_node==NULL)

{

cout<<"------QUEUE OVERFLOW------"<<endl;

}

else

{

if(\*top==NULL && \*Tail==NULL)

{

\*top=new\_node;

\*Tail=new\_node;

new\_node->NEXT=NULL;

}

else

{

(\*Tail)->NEXT=new\_node;

\*Tail=new\_node;

}

}

}

void Dequeue(Node\*\* top,Node\*\* Tail)

{

if(\*top==NULL)

cout<<"------QUEUE UNDERFLOW!------"<<endl;

Node \*temp;

temp=\*top;

if (\*top==\*Tail)

{

\*top=NULL;

\*Tail=NULL;

delete temp;

}

else

{

\*top=temp->NEXT;

delete temp;

}

}

void disfrnt(Node\*\* top)

{

if(\*top==NULL)

cout<<"------EMPTY QUEUE------"<<endl;

else

cout<<(\*top)->info<<endl;

}

void display(Node\*\* top)

{

Node\* temp=\*top;

if(\*top==NULL)

cout<<"------EMPTY QUEUE!------"<<endl;

else

{

cout<<"QUEUE : "<<endl;

while(temp!=NULL)

{

cout<<temp->info<<" ";

temp=temp->NEXT;

}

cout<<endl;

}

}

};

int main()

{

QUEUE \*q1=new QUEUE();

Node\* head;

Node\* tail;

head=q1->sCreateQ();

tail=q1->tCreateQ();

int choice;

int data;

while(1)

{

cout<<endl<<"\*\*\*\*\*\*\*\*Queue Menu\*\*\*\*\*\*\*\*"<<endl;

cout<<"1.enqueue()"<<endl;

cout<<"2.dequeue()"<<endl;

cout<<"3.displayfront()"<<endl;

cout<<"4.displayall()"<<endl;

cout<<"5.exit()"<<endl;

cout<<"Enter Choice(1-5):";

cin>>choice;

switch(choice)

{

case 1:

cout<<"Enter Element to enqueue : ";

cin>>data;

q1->Enqueue(&head,&tail,data);

cout<<endl;

break;

case 2:

q1->Dequeue(&head,&tail);

break;

case 3:

q1->disfrnt(&head);

break;

case 4:

q1->display(&head);

break;

case 5:

break;

default:

cout<<"Enter Valid Choice"<<endl;

break;

}

if(choice==5)

break;

}

return 0;

}

**Q :- WAP to implement priority queue with its basic operations**

*//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*This program is written by Manan Jain(211B173)\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//*

#include <bits/stdc++.h>

using namespace std;

typedef struct node {

int data;

int priority;

struct node\* next;

} Node;

Node\* newNode(int d, int p){

Node\* temp = (Node\*)malloc(sizeof(Node));

temp->data = d;

temp->priority = p;

temp->next = NULL;

return temp;}

int peek(Node\*\* head) { return (\*head)->data; }

void pop(Node\*\* head)

{Node\* temp = \*head;

(\*head) = (\*head)->next;

free(temp);}

void push(Node\*\* head, int d, int p)

{Node\* start = (\*head);

Node\* temp = newNode(d, p);

if ((\*head)->priority < p) {

temp->next = \*head;

(\*head) = temp;

}

else {

while (start->next != NULL

&& start->next->priority > p) {

start = start->next;

}

temp->next = start->next;

start->next = temp;}}

int isEmpty(Node\*\* head) { return (\*head) == NULL; }

int main()

{

Node\* pq = newNode(4, 1);

push(&pq, 5, 2);

push(&pq, 6, 3);

push(&pq, 7, 0);

while (!isEmpty(&pq)) {

cout << " " << peek(&pq);

pop(&pq);}

return 0;}

**LAB:-11(TREES)**

**Q :- WAP to implement priority queue with its basic operations**

*//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*This program is written by Manan Jain(211B173)\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//*

#include <bits/stdc++.h>

using namespace std;

struct Node {

int data;

struct Node \*left, \*right;

Node(int data)

{ this->data = data;

left = right = NULL; }

};

bool isBSTUtil(struct Node\* root, Node\*& prev)

{

if (root) {

if (!isBSTUtil(root->left, prev))

return false;

if (prev != NULL && root->data <= prev->data)

return false;

prev = root;

return isBSTUtil(root->right, prev);

}return true;}

bool isBST(Node\* root){

Node\* prev = NULL;

return isBSTUtil(root, prev);

}int main()

{

struct Node\* root = new Node(3);

root->left = new Node(2);

root->right = new Node(5);

root->left->left = new Node(1);

root->left->right = new Node(4

if (isBST(root))

cout << "Is BST";

else { cout << "Not a BST"; return 0;}