Data Structure Lab: Assignments [Midterm]

Name:Zinniya Taffannum Pritee

Id:20-42050-1

Section: G

**General Instructions:**

1. **Write a C/C++/Java/Python program to implement a QUEUE using 2 Stacks.**

Ans:

#include <bits/stdc++.h>

using namespace std;

struct Queue {

stack<int> s1, s2;

void enQueue(int x)

{

while (!s1.empty()) {

s2.push(s1.top());

s1.pop();

}

s1.push(x);

while (!s2.empty()) {

s1.push(s2.top());

s2.pop();

}

}

int deQueue()

{

if (s1.empty()) {

cout << "Q is Empty";

exit(0);

}

int x = s1.top();

s1.pop();

return x;

}

};

int main()

{

Queue q;

q.enQueue(1);

q.enQueue(2);

q.enQueue(3);

cout << q.deQueue() << '\n';

cout << q.deQueue() << '\n';

cout << q.deQueue() << '\n';

return 0;

}

1. Write a C/C++/Java/Python program to implement one stack efficiently using two queues.

ANS:

**#include <bits/stdc++.h>**

**using namespace std;**

**class Stack {**

**queue<int> q1, q2;**

**int curr\_size;**

**public:**

**Stack()**

**{**

**curr\_size = 0;**

**}**

**void push(int x)**

**{**

**curr\_size++;**

**q2.push(x);**

**while (!q1.empty()) {**

**q2.push(q1.front());**

**q1.pop();**

**}**

**queue<int> q = q1;**

**q1 = q2;**

**q2 = q;**

**}**

**void pop()**

**{**

**if (q1.empty())**

**return;**

**q1.pop();**

**curr\_size--;**

**}**

**int top()**

**{**

**if (q1.empty())**

**return -1;**

**return q1.front();**

**}**

**int size()**

**{**

**return curr\_size;**

**}**

**};**

**int main()**

**{**

**Stack s;**

**s.push(1);**

**s.push(2);**

**s.push(3);**

**cout << "current size: " << s.size()**

**<< endl;**

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

**s.pop();**

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

**s.pop();**

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

**cout << "current size: " << s.size()**

**<< endl;**

**return 0;**

**}**

1. Write a C/C++/Java/Python program to implement two stacks using only one array.

ANS:

|  |
| --- |
|  |

#include <stdlib.h>

using namespace std;

class twoStacks {

int\* arr;

int size;

int top1, top2;

public:

twoStacks(int n)

{

size = n;

arr = new int[n];

top1 = -1;

top2 = size;

} void push1(int x)

{

if (top1 < top2 - 1) {

top1++;

arr[top1] = x;}

else {

cout << "Stack Overflow";

exit(1);

}

}

void push2(int x)

{

if (top1 < top2 - 1) {

top2--;

arr[top2] = x;

}

else {

cout << "Stack Overflow";

exit(1);

}}

int pop1()

{ if (top1 >= 0) {

int x = arr[top1];

top1--;

return x;

}

else {

cout << "Stack UnderFlow";

exit(1);

}}

int pop2()

{if (top2 < size) {

int x = arr[top2];

top2++;

return x;

}

else {

cout << "Stack UnderFlow";

exit(1);}}};

int main()

{twoStacks ts(5);

ts.push1(5);

ts.push2(10);

ts.push2(15);

ts.push1(11);

ts.push2(7);

cout << "Popped element from stack1 is "

<< ts.pop1();

ts.push2(40);

cout << "\nPopped element from stack2 is "

<< ts.pop2(); return 0;

}

1. Write a C/C++/Java/Python program to implement 3 stacks in one array.

#include<bits/stdc++.h>

using namespace std;

class kStacks

{

int \*arr;

int \*top;

int \*next;

int n, k;

int free;

public:

kStacks(int k, int n);

bool isFull() { return (free == -1); }

void push(int item, int sn);

int pop(int sn);

bool isEmpty(int sn) { return (top[sn] == -1); }

};

kStacks::kStacks(int k1, int n1)

{

k = k1, n = n1;

arr = new int[n];

top = new int[k];

next = new int[n];

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

top[i] = -1;

free = 0;

for (int i=0; i<n-1; i++)

next[i] = i+1;

next[n-1] = -1;

}

void kStacks::push(int item, int sn)

{

if (isFull())

{

cout << "\nStack Overflow\n";

return;

}

int i = free;

free = next[i];

next[i] = top[sn];

top[sn] = i;

arr[i] = item;

}

int kStacks::pop(int sn)

{

if (isEmpty(sn))

{

cout << "\nStack Underflow\n";

return INT\_MAX;

}

int i = top[sn];

top[sn] = next[i];

next[i] = free;

free = i;

return arr[i];

}

int main()

{

int k = 3, n = 10;

kStacks ks(k, n);

ks.push(15, 2);

ks.push(45, 2);

ks.push(17, 1);

ks.push(49, 1);

ks.push(39, 1);

ks.push(11, 0);

ks.push(9, 0);

ks.push(7, 0);

cout << "Popped element from stack 2 is " << ks.pop(2) << endl;

cout << "Popped element from stack 1 is " << ks.pop(1) << endl;

cout << "Popped element from stack 0 is " << ks.pop(0) << endl;

return 0;

}