**Laboratory work #4.**

Solve these problems using stack, queue, deque data structures.

Deadline: week 5

<https://informatics.msk.ru/mod/statements/view.php?id=207#1>

#include <iostream>

#include <string>

using namespace std;

#define SIZE 100

struct stack {

int storage[SIZE];

int cursor = 0; // указатель на верхнюю незаполненную ячейку

void push (int n) {

storage[cursor] = n;

cursor++;

}

int pop () {

return storage[--cursor];

}

int back () {

return storage[cursor-1];

}

unsigned int size () {

return cursor;

}

void clear () {

cursor = 0;

}

};

int main () {

stack s;

string str;

int n;

while (cin >> str) {

if (str == "push") {

cin >> n;

s.push(n);

cout << "ok" << endl;

}

else if (str == "pop") cout << s.pop() << endl;

else if (str == "back") cout << s.back() << endl;

else if (str == "size") cout << s.size() << endl;

else if (str == "clear") {

s.clear();

cout << "ok" << endl;

}

else if (str == "exit") {

cout << "bye" << endl;

break;

}

}

return 0;

}

<https://informatics.msk.ru/mod/statements/view.php?id=207&chapterid=55#1>

#include <iostream>

using namespace std;

struct stack { //создаем структуру

long long s[100];

int cursor=0; //указатель

void push(long long n){ //задаем функцию для каждой из команд: push n, pop, back, size, clear

s[cursor++]=n;

}

long long pop(){

return s[--cursor];

}

long long back(){

return s[cursor-1];

}

unsigned size(){

return cursor;

}

void clear(){

cursor=0;

s[cursor]=0;

}

};

int main() {

stack x;

string b; //задаем строку

long long n;

while(cin>>b){ //читаем ее

if(b=="push"){ //выполняем команды, которые встречаем в строке

cin>>n;

x.push(n);

cout<<"ok"<<endl;

}

else if(b=="pop"){

if(x.size()==0){

cout<<"error"<<endl;

}

else

cout<<x.pop()<<endl;

}

else if(b=="back"){

if(x.size()==0){

cout<<"error"<<endl;

}

else cout<<x.back()<<endl;

}

else if(b=="size"){

cout<<x.size()<<endl;

}

else if(b=="clear"){

x.clear();

cout<<"ok"<<endl;

}

else if(b=="exit"){

cout<<"bye"<<endl;

break;

}

}

return 0;

}

<https://informatics.msk.ru/mod/statements/view.php?id=207&chapterid=57#1>

import java.util.\*;

import java.lang.\*;

import java.io.\*;

class MyQueue {

int[] storage = new int[100];

int start;

int finish;

MyQueue() {

start = 0;

finish = 0;

}

void push(int n) {

storage[finish] = n;

finish++;

}

int pop() {

int a = storage[start];

start++;

return a;

}

int front() {

return storage[start];

}

int size() {

return finish - start;

}

String clear() {

finish = 0;

start = 0;

return "ok";

}

String exit() {

return "bye";

}

}

public class Main {

public static void main(String[] args) throws java.lang.Exception {

String a;

MyQueue x = new MyQueue();

try (Scanner sc = new Scanner(System.in)) {

while (sc.hasNextLine()){

a = sc.next();

if (a.equals("push")) {

int n;

n = sc.nextInt();

x.push(n);

System.out.println("ok");

}

if (a.equals("pop")) {

System.out.println(x.pop());

}

if (a.equals("front")) {

System.out.println(x.front());

}

if (a.equals("size")) {

System.out.println(x.size());

}

if (a.equals("clear")) {

System.out.println(x.clear());

}

if (a.equals("exit")) {

System.out.println(x.exit());

break;

}

}

}

}

}

<https://informatics.msk.ru/mod/statements/view.php?id=207&chapterid=58#1>

<https://informatics.msk.ru/mod/statements/view.php?id=207&chapterid=60#1>

#include <iostream>

#include <string>

using namespace std;

struct stack{

int storage[1000];

int end=0;

int sizze=0;

int start=1;

int lol=100;

int push\_back(int x){

end = (end + 1)%lol;

storage[end] = x;

sizze++;

}

int push\_front(int x){start = (start - 1+lol)%lol;

storage[start] = x;

sizze++;}

void pop\_back(){end = (end - 1+lol)%lol;

sizze--;}

void pop\_front(){start = (start + 1)%lol;

sizze--;}

int back() const{return storage[end];}

int size() const{return sizze;}

void clear(){end = 0;

sizze = 0;

start = 1;

}

int front() const{

return storage[start];

}

};

int main() {

stack storage;

string s;

int n;

while(cin >> s){

if(s == "push\_back"){

cin >> n;

storage.push\_back(n);

cout << "ok\n";

}else if(s == "push\_front"){

cin >> n;

storage.push\_front(n);

cout << "ok\n";

}

else if(s == "pop\_back"){

if(storage.size()){

cout << storage.back() << endl;

storage.pop\_back();

}else{

cout << "error\n";

}

}else if(s == "pop\_front"){

if(storage.size()){

cout << storage.front() << endl;

storage.pop\_front();

}else{

cout << "error\n";

}

}else if(s == "front"){

if(storage.size()){

cout << storage.front() << endl;

}

else{

cout << "error\n";

}

}

else if(s == "back"){

if(storage.size()){

cout << storage.back() << endl;

}else{

cout << "error\n";

}

}else if(s == "size"){

cout << storage.size() << endl;

}else if(s == "clear"){

storage.clear();

cout << "ok\n";

}else if(s == "exit"){

cout << "bye\n";

return 0;

}

}

return 0;

}

<https://informatics.msk.ru/mod/statements/view.php?id=207&chapterid=61#1>

<https://informatics.msk.ru/mod/statements/view.php?id=206#1>

<https://informatics.msk.ru/mod/statements/view.php?id=206&chapterid=50#1>

<https://informatics.msk.ru/mod/statements/view.php?id=206&chapterid=112984#1>

<https://informatics.msk.ru/mod/statements/view.php?id=206&chapterid=53#1>

<https://leetcode.com/problems/valid-parentheses/>

class Solution {

public boolean isValid(String s) {

boolean result = true;

Stack<Character> checking = new Stack<Character>();

for(char c:s.toCharArray()){

if(c=='('){

checking.push(')');

}

else if(c=='{'){

checking.push('}');

}

else if(c=='['){

checking.push(']');

}

else if(checking.isEmpty()||checking.pop()!=c){

result = false;

}

}

if (result){

result = checking.isEmpty();

}

return result;

}

}

<https://leetcode.com/problems/min-stack/>

class MinStack {

/\*\* initialize your data structure here. \*/

private int min=Integer.MAX\_VALUE;

private ArrayList<Integer>al=new ArrayList<Integer>();

public MinStack()

{

}

public void push(int x) {

if(al.size()!=0)

{

al.add(0,x);

return;

}

else{

al.add(x);

return;

}

}

public void pop()

{

min=Integer.MAX\_VALUE;

if(al.size()==0)

{

return;

}

else{

al.remove(0);

}

}

public int top()

{

if(al.size()==0)

{

return -1;

}

return al.get(0);

}

public int getMin()

{

for(int i:al)

{

if(i<min)

{

min=i;

}

}

return min;

}

}

<https://leetcode.com/problems/backspace-string-compare/>

class Solution {

public boolean backspaceCompare(String S, String T) {

int i = S.length()-1, j = T.length()-1, del1 = 0, del2 = 0;

while(j>=0 || i>=0){

if(i>=0 && S.charAt(i)=='#' || j>=0 && T.charAt(j)=='#'){

while(i>=0 && S.charAt(i) == '#'){

del1++;

i--;

}

while(j>=0 && T.charAt(j) == '#'){

del2++;

j--;

}

while(i>=0 && S.charAt(i)!='#' && del1!=0){

i--;

del1--;

}

while(j>=0 && T.charAt(j)!='#' && del2!=0){

j--;

del2--;

}

}

else{

//case where one index is less than 0 and one is not

if(i<0 ^ j<0 || S.charAt(i)!=T.charAt(j)){

return false;

}

i = (i>=0)?i-1:i;

j = (j>=0)?j-1:j;

}

}

return true;

}

}

<https://leetcode.com/problems/evaluate-reverse-polish-notation/>

class Solution {

public int evalRPN(String[] tokens)

{

Stack<Integer> stack=new Stack<>();

for(String s:tokens)

{

if(isDigit(s))

{

stack.push(Integer.parseInt(s));

}

else

{

int b=stack.pop();

int a=stack.pop();

stack.push(calc(s,a,b));

}

}

return stack.pop();

}

public boolean isDigit(String s)

{

if(s.equals("+")||s.equals("/")||s.equals("\*")||s.equals("-"))

{

return false;

}

return true;

}

public int calc(String op,int a,int b)

{

switch(op)

{

case "+": return a+b;

case "-": return a-b;

case "\*": return a\*b;

case "/": return a/b;

}

return 1;

}

}