Белорусский государственный технологический университет

Факультет информационных технологий

Кафедра программной инженерии

 Лабораторная работа 7

По дисциплине «Основы алгоритмизации программирования»

На тему «Полустатические структуры данных: стеки»

Выполнил:

Студент 1 курса 6 группы

Кучерук Николай Петрович

Преподаватель: асс. Андронова М.В.

2023, Минск

**Задание**

| **№ варианта** | **Условие задачи** |
| --- | --- |
| **9** | Разработать функцию, которая формирует стек **Stack**, включив в него по одному разу элементы, которые входят в один из стеков **Stack1** и **Stack2**, но не входят в другой. |

|  |
| --- |
| **Код программы** |
| **Содержание main.cpp**  #include "stack.h"  int main()  {  system("color 70");  setlocale(LC\_CTYPE, "Russian");  stack first, second, third;//создаем стэк  int choice;  int value;  menu(); // вывести меню  cout << " ? ";  cin >> choice;  while (choice != 8)  {  switch (choice)  {  case 1: cout << "Введите число " << endl; // добавить число в стэк 1  cin >> value;  push\_front(value, first);  cout << "Cтэк: ";  output(first);  break;  case 2:  {  pop\_front(first);  cout << "Cтэк: ";  output(first);  }  break;  case 3: cout << "Введите число " << endl; // добавить число в стэк 2  cin >> value;  push\_front(value, second);  cout << "Cтэк: ";  output(second);  break;  case 4:  {  pop\_front(second);  cout << "Cтэк: ";  output(second);  break;  }  case 5:  {  filewr(first);  cout << "Файл записан " << endl;  break;  }  case 6:  {  filerd(first);  cout << "Данные из файла переписаны в стэк " << endl;  output(first);  break;  }  case 7:  {  merge(first, second, third);  cout << "Полученный стэк: ";  output(third);  break;  }  default: cout << "Неправильный выбор" << endl;  menu();  break;  }  cout << "? ";  cin >> choice;  }  cout << "Конец" << endl;  clear(first);  return 0;  }  **Содержание stack.h**  #pragma once  #include <iostream>  #include <fstream>  using namespace std;  struct node//элемент стэка  {  int data;//информация, хранящаяся в элементе  node\* next;//указатель на следующий элемент стэка  node(int data) {//конструктор элемента  this->data = data;  this->next = NULL;  }  };  struct stack//структура стэка на основе списка  {  node\* head, \* tail;//указатели на первый и последний элемент стэка  stack() {//конструктор стэка  this->head = this->tail = NULL;  }  };  void menu(void); //функция, показывающая меню  void filewr(stack&); //функция записи стэка в файл  void filerd(stack&); //функция получения стэка из файла  void output(stack); //функция вывода стэка  void push\_front(int,stack&); //функция для добавки элемента в стэк  void pop\_front(stack&);//функция для удаления элемента стэка,добавленного последним  void clear(stack&);//функция очистки стэка  void merge(stack, stack, stack&);//функция для получения нового уникального стэка  **Содержание func\_stack.cpp**  #include "stack.h"  void menu(void) //Вывод меню  {  cout << "Сделайте выбор:" << endl;  cout << " 1 - Ввод числа" << endl;  cout << " 2 - Удаление элемента" << endl;  cout << " 3 - Ввод числа во второй стэк" << endl;  cout << " 4 - Удаление элемента из второго стэка" << endl;  cout << " 5 - Ввод в файл первого стэка" << endl;  cout << " 6 - Вывод из файла в первый стэк" << endl;  cout << " 7 - Получение нового стэка из уникальных элементов первого стэка" << endl;  cout << " 8 - Выход" << endl;  }  void filewr(stack& p) {//запись данных стэка в файл  ofstream file("list.txt");  for (node\* Node = p.head; Node != NULL; Node = Node->next) {  file << Node->data << " ";  }  file.close();  }  void filerd(stack& temp) {//чтение данных из файла в стэк  ifstream file("list.txt");  if (file.fail())  {  cout << "\n File cannot be opened";  exit(1);  }  clear(temp);  int a;  while (file >> a) {  push\_front(a, temp);  }  file.close();  }  void output(stack temp) {//вывод стэка  for (node\* Node = temp.head; Node != NULL; Node = Node->next) {  cout << Node->data << " ";  }  cout << endl;  }  void push\_front(int data, stack& Stack) {//вхождение элемента в стэк  node\* Node = new node(data);  Node->next = Stack.head;  Stack.head = Node;  if (Stack.tail == NULL)Stack.tail = Node;  }  void pop\_front(stack& Stack) {//удаляем последний вошешдший элемент  if (Stack.head == NULL) return;  if (Stack.head == Stack.tail) {  delete Stack.tail;  Stack.head = Stack.tail = NULL;  return;  }  node\* node = Stack.head;  Stack.head = node->next;  delete node;  }  void clear(stack& Stack) {//функция очистки стэка  while (Stack.head != NULL) pop\_front(Stack);  }  void merge(stack first, stack second, stack& third) {//функция объединения  bool uniq = 1;  for (node\* Node = first.head; Node != NULL; Node = Node->next) {  for (node\* Node2 = second.head; Node2 != NULL; Node2 = Node2->next) {  if (Node2->data == Node->data) { uniq = 0; }  }  if (uniq) { push\_front(Node->data, third); }  uniq = 1;  }  } |
| **Результат программы** |
|  |

Дополнительные задания

| **№ варианта** | **Условие задачи** |
| --- | --- |
| **2** | Разработать функцию, которая удаляет из стека первый отрицательный элемент, если такой есть. |

|  |
| --- |
| **Код программы** |
| **Содержание main.cpp**  #include "stack.h"  int main()  {  system("color 70");  setlocale(LC\_CTYPE, "Russian");  stack first;//создаем стэк  int choice;  int value;  menu(); // вывести меню  cout << " ? ";  cin >> choice;  while (choice != 6)  {  switch (choice)  {  case 1: cout << "Введите число " << endl; // добавить число в стэк  cin >> value;  push\_front(value, first);  cout << "Cтэк: ";  output(first);  break;  case 2:  {  pop\_front(first);  cout << "Cтэк: ";  output(first);  }  break;  case 3:  {  filewr(first);  cout << "Файл записан " << endl;  break;  }  case 4:  {  filerd(first);  cout << "Данные из файла переписаны в стэк " << endl;  output(first);  break;  }  case 5:  {  delete\_neg(first);  cout << "Полученный стэк: ";  output(first);  break;  }  default: cout << "Неправильный выбор" << endl;  menu();  break;  }  cout << "? ";  cin >> choice;  }  cout << "Конец" << endl;  clear(first);  return 0;  }  **Содержание stack.h**  #pragma once  #include <iostream>  #include <fstream>  using namespace std;  struct node//элемент стэка  {  int data;//информация, хранящаяся в элементе  node\* next;//указатель на следующий элемент стэка  node(int data) {//конструктор элемента  this->data = data;  this->next = NULL;  }  };  struct stack//структура стэка на основе списка  {  node\* head, \* tail;//указатели на первый и последний элемент стэка  stack() {//конструктор стэка  this->head = this->tail = NULL;  }  };  void menu(void); //функция, показывающая меню  void filewr(stack&); //функция записи стэка в файл  void filerd(stack&); //функция получения стэка из файла  void output(stack); //функция вывода стэка  void push\_front(int,stack&); //функция для добавки элемента в стэк  void pop\_front(stack&);//функция для удаления элемента стэка,добавленного последним  void clear(stack&);//функция очистки стэка  void delete\_neg(stack&);//функция для удаления первого отрицательного элемента  **Содержание func\_stack.cpp**  #include "stack.h"  void menu(void) //Вывод меню  {  cout << "Сделайте выбор:" << endl;  cout << " 1 - Ввод числа" << endl;  cout << " 2 - Удаление элемента" << endl;  cout << " 3 - Ввод в файл первого стэка" << endl;  cout << " 4 - Вывод из файла в первый стэк" << endl;  cout << " 5 - Удаление первого отрицательного элемента" << endl;  cout << " 6 - Выход" << endl;  }  void filewr(stack& p) {//запись данных стэка в файл  ofstream file("list.txt");  for (node\* Node = p.head; Node != NULL; Node = Node->next) {  file << Node->data << " ";  }  file.close();  }  void filerd(stack& temp) {//чтение данных из файла в стэк  ifstream file("list.txt");  if (file.fail())  {  cout << "\n File cannot be opened";  exit(1);  }  clear(temp);  int a;  while (file >> a) {  push\_front(a, temp);  }  file.close();  }  void output(stack temp) {//вывод стэка  for (node\* Node = temp.head; Node != NULL; Node = Node->next) {  cout << Node->data << " ";  }  cout << endl;  }  void push\_front(int data, stack& Stack) {//вхождение элемента в стэк  node\* Node = new node(data);  Node->next = Stack.head;  Stack.head = Node;  if (Stack.tail == NULL)Stack.tail = Node;  }  void pop\_front(stack& Stack) {//удаляем последний вошешдший элемент  if (Stack.head == NULL) return;  if (Stack.head == Stack.tail) {  delete Stack.tail;  Stack.head = Stack.tail = NULL;  return;  }  node\* node = Stack.head;  Stack.head = node->next;  delete node;  }  void clear(stack& Stack) {//функция очистки стэка  while (Stack.head != NULL) pop\_front(Stack);  }  void delete\_neg(stack& first) {//функция удаления первого отрицательного элемента  node\* Node = first.head;  node\* Node\_pr=Node;  while (Node != NULL) {  if (Node->data < 0) {  if (Node == first.head) { pop\_front(first); return; }  if (Node->next==NULL) { first.tail = Node\_pr;}  Node\_pr->next = Node->next;  delete Node;  return;  }  Node = Node->next;  Node\_pr->next = Node;  }  } |
| **Результат программы** |
|  |

| **№ варианта** | **Условие задачи** |
| --- | --- |
| **3** | Разработать функцию, которая удаляет первый положительный элемент, если такой есть. |

|  |
| --- |
| **Код программы** |
| **Содержание main.cpp**  #include "stack.h"  int main()  {  system("color 70");  setlocale(LC\_CTYPE, "Russian");  stack first;//создаем стэк  int choice;  int value;  menu(); // вывести меню  cout << " ? ";  cin >> choice;  while (choice != 6)  {  switch (choice)  {  case 1: cout << "Введите число " << endl; // добавить число в стэк  cin >> value;  push\_front(value, first);  cout << "Cтэк: ";  output(first);  break;  case 2:  {  pop\_front(first);  cout << "Cтэк: ";  output(first);  }  break;  case 3:  {  filewr(first);  cout << "Файл записан " << endl;  break;  }  case 4:  {  filerd(first);  cout << "Данные из файла переписаны в стэк " << endl;  output(first);  break;  }  case 5:  {  delete\_pos(first);  cout << "Полученный стэк: ";  output(first);  break;  }  default: cout << "Неправильный выбор" << endl;  menu();  break;  }  cout << "? ";  cin >> choice;  }  cout << "Конец" << endl;  clear(first);  return 0;  }  **Содержание stack.h**  #pragma once  #include <iostream>  #include <fstream>  using namespace std;  struct node//элемент стэка  {  int data;//информация, хранящаяся в элементе  node\* next;//указатель на следующий элемент стэка  node(int data) {//конструктор элемента  this->data = data;  this->next = NULL;  }  };  struct stack//структура стэка на основе списка  {  node\* head, \* tail;//указатели на первый и последний элемент стэка  stack() {//конструктор стэка  this->head = this->tail = NULL;  }  };  void menu(void); //функция, показывающая меню  void filewr(stack&); //функция записи стэка в файл  void filerd(stack&); //функция получения стэка из файла  void output(stack); //функция вывода стэка  void push\_front(int,stack&); //функция для добавки элемента в стэк  void pop\_front(stack&);//функция для удаления элемента стэка,добавленного последним  void clear(stack&);//функция очистки стэка  void delete\_pos(stack&);//функция удаления первого положительного элемента  **Содержание func\_stack.cpp**  #include "stack.h"  void menu(void) //Вывод меню  {  cout << "Сделайте выбор:" << endl;  cout << " 1 - Ввод числа" << endl;  cout << " 2 - Удаление элемента" << endl;  cout << " 3 - Ввод в файл первого стэка" << endl;  cout << " 4 - Вывод из файла в первый стэк" << endl;  cout << " 5 - Удаление первого положительного элемента" << endl;  cout << " 6 - Выход" << endl;  }  void filewr(stack& p) {//запись данных стэка в файл  ofstream file("list.txt");  for (node\* Node = p.head; Node != NULL; Node = Node->next) {  file << Node->data << " ";  }  file.close();  }  void filerd(stack& temp) {//чтение данных из файла в стэк  ifstream file("list.txt");  if (file.fail())  {  cout << "\n File cannot be opened";  exit(1);  }  clear(temp);  int a;  while (file >> a) {  push\_front(a, temp);  }  file.close();  }  void output(stack temp) {//вывод стэка  for (node\* Node = temp.head; Node != NULL; Node = Node->next) {  cout << Node->data << " ";  }  cout << endl;  }  void push\_front(int data, stack& Stack) {//вхождение элемента в стэк  node\* Node = new node(data);  Node->next = Stack.head;  Stack.head = Node;  if (Stack.tail == NULL)Stack.tail = Node;  }  void pop\_front(stack& Stack) {//удаляем последний вошешдший элемент  if (Stack.head == NULL) return;  if (Stack.head == Stack.tail) {  delete Stack.tail;  Stack.head = Stack.tail = NULL;  return;  }  node\* node = Stack.head;  Stack.head = node->next;  delete node;  }  void clear(stack& Stack) {//функция очистки стэка  while (Stack.head != NULL) pop\_front(Stack);  }  void delete\_pos(stack& first) {//функция удаления первого положительного элемента  node\* Node = first.head;  node\* Node\_pr=Node;  while (Node != NULL) {  if (Node->data > 0) {  if (Node == first.head) { pop\_front(first); return; }  if (Node->next==NULL) { first.tail = Node\_pr;}  Node\_pr->next = Node->next;  delete Node;  return;  }  Node = Node->next;  Node\_pr->next = Node;  }  } |
| **Результат программы** |
|  |

| **№ варианта** | **Условие задачи** |
| --- | --- |
| **5** | Разработать функцию, которая формирует стек **Stack**, включив в него по одному разу элементы, которые входят в стек **Stack1**, но не входят в стек **Stack2**. |

|  |
| --- |
| **Код программы** |
| **Содержание main.cpp**  #include "stack.h"  int main()  {  system("color 70");  setlocale(LC\_CTYPE, "Russian");  stack first, second, third;//создаем список  int choice;  int value;  menu(); // вывести меню  cout << " ? ";  cin >> choice;  while (choice != 8)  {  switch (choice)  {  case 1: cout << "Введите число " << endl; // добавить число в стэк 1  cin >> value;  push\_front(value, first);  cout << "Cтэк: ";  output(first);  break;  case 2:  {  pop\_front(first);  cout << "Cтэк: ";  output(first);  }  break;  case 3: cout << "Введите число " << endl; // добавить число в стэк 2  cin >> value;  push\_front(value, second);  cout << "Cтэк: ";  output(second);  break;  case 4:  {  pop\_front(second);  cout << "Cтэк: ";  output(second);  break;  }  case 5:  {  filewr(first);  cout << "Файл записан " << endl;  break;  }  case 6:  {  filerd(first);  cout << "Данные из файла переписаны в стэк " << endl;  output(first);  break;  }  case 7:  {  merge(first, second, third);  cout << "Полученный стэк: ";  output(third);  break;  }  default: cout << "Неправильный выбор" << endl;  menu();  break;  }  cout << "? ";  cin >> choice;  }  cout << "Конец" << endl;  clear(first);  return 0;  }  **Содержание stack.h**  #pragma once  #include <iostream>  #include <fstream>  using namespace std;  struct node//элемент стэка  {  int data;//информация, хранящаяся в элементе  node\* next;//указатель на следующий элемент стэка  node(int data) {//конструктор элемента  this->data = data;  this->next = NULL;  }  };  struct stack//структура стэка на основе списка  {  node\* head, \* tail;//указатели на первый и последний элемент стэка  stack() {//конструктор стэка  this->head = this->tail = NULL;  }  };  void menu(void); //функция, показывающая меню  void filewr(stack&); //функция записи стэка в файл  void filerd(stack&); //функция получения стэка из файла  void output(stack); //функция вывода стэка  void push\_front(int,stack&); //функция для добавки элемента в стэк  void pop\_front(stack&);//функция для удаления элемента стэка,добавленного последним  void clear(stack&);//функция очистки стэка  void merge(stack, stack, stack&);//функция для получения нового уникального стэка  **Содержание func\_stack.cpp**  #include "stack.h"  void menu(void) //Вывод меню  {  cout << "Сделайте выбор:" << endl;  cout << " 1 - Ввод числа" << endl;  cout << " 2 - Удаление элемента" << endl;  cout << " 3 - Ввод числа во второй стэк" << endl;  cout << " 4 - Удаление элемента из второго стэка" << endl;  cout << " 5 - Ввод в файл первого стэка" << endl;  cout << " 6 - Вывод из файла в первый стэк" << endl;  cout << " 7 - Получение нового стэка из уникальных элементов первого стэка" << endl;  cout << " 8 - Выход" << endl;  }  void filewr(stack& p) {//запись данных стэка в файл  ofstream file("list.txt");  for (node\* Node = p.head; Node != NULL; Node = Node->next) {  file << Node->data << " ";  }  file.close();  }  void filerd(stack& temp) {//чтение данных из файла в стэк  ifstream file("list.txt");  if (file.fail())  {  cout << "\n File cannot be opened";  exit(1);  }  clear(temp);  int a;  while (file >> a) {  push\_front(a, temp);  }  file.close();  }  void output(stack temp) {//вывод стэка  for (node\* Node = temp.head; Node != NULL; Node = Node->next) {  cout << Node->data << " ";  }  cout << endl;  }  void push\_front(int data, stack& Stack) {//вхождение элемента в стэк  node\* Node = new node(data);  Node->next = Stack.head;  Stack.head = Node;  if (Stack.tail == NULL)Stack.tail = Node;  }  void pop\_front(stack& Stack) {//удаляем последний вошешдший элемент  if (Stack.head == NULL) return;  if (Stack.head == Stack.tail) {  delete Stack.tail;  Stack.head = Stack.tail = NULL;  return;  }  node\* node = Stack.head;  Stack.head = node->next;  delete node;  }  void clear(stack& Stack) {//функция очистки стэка  while (Stack.head != NULL) pop\_front(Stack);  }  void merge(stack first, stack second, stack& third) {//функция объединения  bool uniq = 1;  for (node\* Node = first.head; Node != NULL; Node = Node->next) {  for (node\* Node2 = second.head; Node2 != NULL; Node2 = Node2->next) {  if (Node2->data == Node->data) { uniq = 0; }  }  if (uniq) { push\_front(Node->data, third); }  uniq = 1;  }  } |
| **Результат программы** |
|  |