Беларускі дзяржаўны тэхналагічны ўніверсітэт

Факультэт інфармацыйных тэхналогій

Кафедра праграмнай інжэнерыі

Лабараторная работа №8

Па дысцыпліне «Асновы алгарытмізацыі і праграмавання»

На тэму «Чарга»

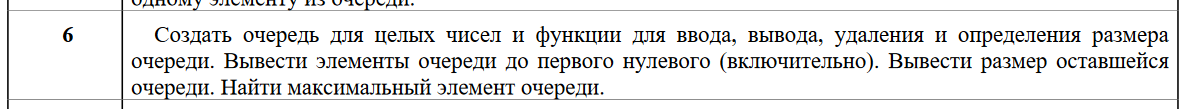
Выканала:

Студэнтка 1 курса 6 группы

Жучкевіч Кацярына Сяргееўна

Выкладчык: асс. Андронава М.В.

Мінск, 2024

****

#include <iostream>

#include <Windows.h>

using namespace std;

struct Que {//структура для чаргі

int data;

Que\* next;

};

struct Queue {//сама чарга

Que\* start; //паказальнік для пачатка чаргі

Que\* end; // паказальнік для канчатка чаргі

int size;

};

int countAll = 0;

int countSmall = 0;

//прататыпы

bool isEmpty(const Queue& queue);

void addQueue(Queue& queue, int value);

void delQueue(Queue& queue);

void showQueue(const Queue& queue);

void Task(Queue& queue);

int findMax(Queue& queue);

int main() {

SetConsoleOutputCP(1251);//гэта каб не змяняць заўсёды беларускую і на англійскую :\_(

Queue queue;//ствараем пустую чаргу

queue.start = nullptr;

queue.end = nullptr;

queue.size = 0;

int maxSize;

cout << "Увядзiце максімальны памер чаргi: ";

cin >> maxSize;

int choice;//меню

cout << "\nМеню:" << endl;

cout << "1) Дадаць элемент ў чаргу" << endl;

cout << "2) Выдаліць элемент з чаргі" << endl;

cout << "3) Вывесці чаргу" << endl;

cout << "4) Паказаць памер чаргі" << endl;

cout << "5) Заданне" << endl;

cout << "6) Знайсці максімальны элемент" << endl;

cout << "0) Выхад" << endl;

do {

cout << "Увядзіце: ";

cin >> choice;

switch (choice) {

case 1: {

if (queue.size >= maxSize) {

cout << "Месца ў чарзе больш няма." << endl;

}

else {

int value;

cout << "Увядзіце элемент: ";

cin >> value;

addQueue(queue, value);

countAll++;

cout << "Элемент даданы ў чаргу" << endl;

}

break;

}

case 2: {

if (isEmpty(queue)) {

std::cout << "Не магу нічога зрабіць, бо чарга пустая((" << std::endl;

}

else {

delQueue(queue);

countAll--;

cout << "Элемент выдалены!" << endl;

}

break;

}

case 3: {

showQueue(queue);

break;

}

case 4: {

cout << "Памер чаргі: " << countAll - countSmall << endl;//лічым, колькі элементаў засталося ў чарзе

break;

}

case 5: {

Task(queue);

break;

}

case 6: {

cout << findMax(queue) << endl;

break;

}

case 0: {

cout << "Усё!" << endl;

break;

}

default: {

cout << "Гэты варыянт дрэнны(" << endl;

break;

}

}

} while (choice != 0);

return 0;

}

bool isEmpty(const Queue& queue) { //праверка, каб чарга была пустая

return queue.start == nullptr;

}

void addQueue(Queue& queue, int value) {//функцыя для таго, каб дадаць элемент ў чаргу

Que\* newNode = new Que;

newNode->data = value;

newNode->next = nullptr;

if (isEmpty(queue)) {

queue.start = newNode;

queue.end = newNode;

}

else {

queue.end->next = newNode;

queue.end = newNode;

}

queue.size++;

}

void delQueue(Queue& queue) { //функцыя, каб выдаліць элемент з чаргі

if (isEmpty(queue)) {//праглядаем, каб чарга не была пустой

cout << "Не магу нічога зрабіць, бо чарга пустая((" << endl;

return;

}

Que\* temp = queue.start;

queue.start = queue.start->next;

delete temp;

if (queue.start == nullptr) {

queue.end = nullptr;

}

queue.size--;

}

void showQueue(const Queue& queue) {//вывад чаргі

if (isEmpty(queue)) {

cout << "Чарга пустая" << std::endl;

return;

}

Que\* current = queue.start;

cout << "Вось элементы чаргі: ";

while (current != nullptr) {

cout << current->data << " ";

current = current->next;

}

cout << endl;

}

int findMax(Queue& queue) { //функцфя для такго, каб знайсці максімальны элемент

Que\* current = queue.start;

int max\_value = current->data;

while (current != nullptr) {

if (current->data > max\_value) {

max\_value = current->data;

}

current = current->next;

}

return max\_value;

}

void Task(Queue& queue) {//функцыя з асноўным заданнем

if (isEmpty(queue)) {

cout << "Не магу нічога зрабіць, бо чарга пустая((" << endl;

return;

}

Que\* current = queue.start;

Que\* prev = nullptr;

int count = 0;

while (current != nullptr) {

cout << current->data << " ";//выводзім да нуля

countSmall++;//лічым

if (current->data == 0) {

return;

}

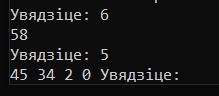
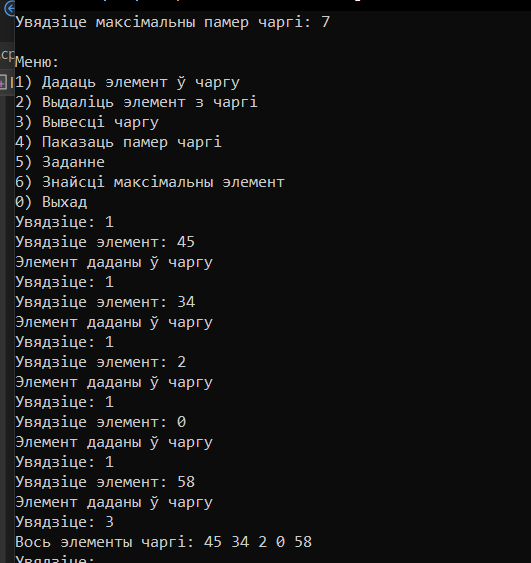
prev = current;//змяшчаем паказальнік

current = current->next;

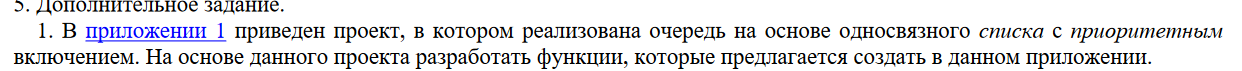
}

delete current;

}

****

**Дадатковыя варыянты:**

****

#include<iostream>

using namespace std;

struct Item

{

int data;

Item\* next;

};

Item\* head, \* tail;

bool isNull(void) //Проверка на пустоту

{

return (head == NULL);

}

void deletFirst() //Извлечение элемента из начала

{

if (isNull())

cout << "Очередь пуста" << endl;

else

{

Item\* p = head;

head = head->next;

delete p;

}

}

void getFromHead() //Получение элемента из начала

{

if (isNull())

cout << "Очередь пуста" << endl;

else

cout << "Начало = " << head->data << endl;

}

void insertToQueue(int x) //Добавление элемента в очередь

{

Item\* p = new Item; //новый указатель

p->data = x;

p->next = NULL;

Item\* v = new Item; //указатель для нового числа

Item\* p1 = new Item;

Item\* p2 = new Item;

int i = 0; //флажок

if (isNull())

head = tail = p;

else

{

p2 = head; p1 = head;

while (p1 != NULL) //пока очередь не закончится

{

if (i == 1)

{

if (x <= p1->data) //число меньше, чем в очереди

{

v->data = x;

v->next = p1;

p2->next = v;

return;

}

p2 = p2->next; // следующее число

}

else

{

if (x <= p1->data){

v->data = x;

v->next = p1;

head = v;

return;

}

}

p1 = p1->next;

i = 1;

}

if (p1 == NULL)

{

tail->next = p;

tail = p;

}

}

}

void printQueue() //Вывод очереди

{

Item\* p = new Item;

if (isNull())

cout << "Очередь пуста" << endl;

else

{

cout << "Очередь = ";

p = head;

while (!isNull())

{

if (p != NULL)

{

cout << p->data << " "; cout << "->";

p = p->next;

}

else

{

cout << "NULL" << endl;

return;

}

}

}

}

void clrQueue() //Очистка очереди

{

while (!isNull()) deletFirst();

}

void inputQueuePrio(int x, bool isFIFO)

{

if (isNull() || x >= head->data) {

insertToQueue(x);

}

else {

Item\* p = new Item;

p->data = x;

p->next = head;

head = p;

}

if (!isFIFO) {

Item\* p = head;

while (p->next->next != nullptr)

{

p = p->next;

}

delete p->next;

p->next = nullptr;

}

}

void getNewQueue(bool isFIFO)

{

if (isNull())

cout << "Чарга пустая!!!" << endl;

else {

if (isFIFO)

getFromHead();

else {

Item\* p = head;

while (p->next != nullptr) {

p = p->next;

}

cout << "Пачатак: " << p->data << endl;

}

}

}

int main()

{

setlocale(LC\_CTYPE, "Russian");

int i = 1, choice = 1, z; head = NULL; tail = NULL;

while (choice != 0)

{

cout << "1 - добавить элемент" << endl;

cout << "2 - получить элемент с начала" << endl;

cout << "3 - извлечь элемент с начала" << endl;

cout << "4 - вывести элементы" << endl;

cout << "5 - очистить очередь" << endl;

cout << "0 - выход" << endl;

cout << "Выберите действие "; cin >> choice;

switch (choice)

{

case 1: cout << "Введите элемент: "; cin >> z;

insertToQueue(z); printQueue(); break;

case 2: getFromHead(); break;

case 3: deletFirst(); break;

case 4: printQueue(); break;

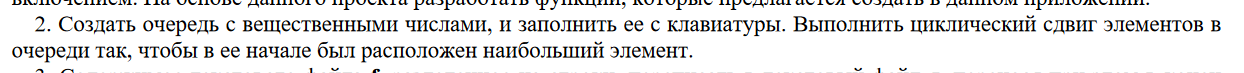
case 5: clrQueue(); break;

}

}

return 0;

}



#include <iostream>

#include <queue>//падлучэнне бібліятэкі для працы з чаргой

#include <Windows.h>

using namespace std;

int main() {

SetConsoleOutputCP(1251);

queue<int> Queue;//ствараем вектар

int size;

int maxElement = -1;

//Ввод элементов

cout << "Увядзіце колькасць элементаў: ";

cin >> size;

cout << "Увядзіце элементы:\n";

for (int i = 0; i < size; ++i) {

int queueElement;

cin >> queueElement;

Queue.push(queueElement);//дадаем элемент ў канчатак чаргі

if (queueElement > maxElement) {

maxElement = queueElement;

}

}

queue<int> temp;

while (!Queue.empty()) {//пакуль чарга пустая

if (Queue.front() == maxElement) {

break;

}

temp.push(Queue.front());//змяняем месцамі элементы

Queue.pop();

}

while (!temp.empty()) {

Queue.push(temp.front());

temp.pop();

}

cout << "Адноўленая чарга:\n";//вывад новай чаргі

while (!Queue.empty()) {

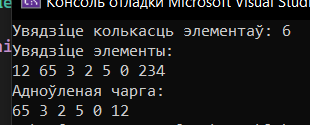
std::cout << Queue.front() << " ";

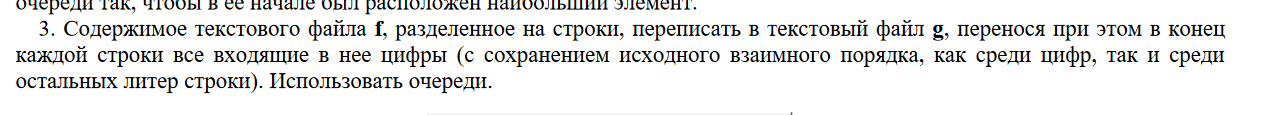
Queue.pop();

}

return 0;

}

 // адзін атрымаўся лішнім



#include <iostream>//падлучаем бібліятэкі

#include <fstream>

#include <queue>

#include <string>

using namespace std;

int main() {

setlocale(0, "");

ifstream fin("f.txt");//адчыняем файл для запісу

ofstream fout("g.txt");//для вывада

if (!fin.is\_open() || !fout.is\_open()) {//праглядаем прысутнасць файлаў

cout << "Немагчыма адчыніць файлы!!";

return 1;

}

string line;

while (getline(fin, line)) {//пакуль не канчатак

queue<char> nums;

for (char c : line) {//цыкл "для кожнага"

if (isdigit(c))

nums.push(c);

else

fout << c;

}

while (!nums.empty()) {

fout << nums.front();

nums.pop();

}

fout << endl;

}

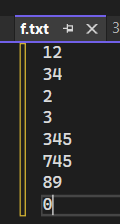
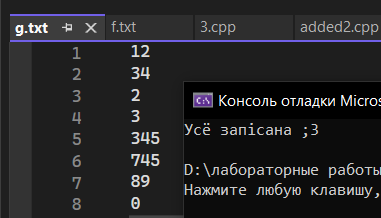
fin.close();//зачыняем файлы

fout.close();

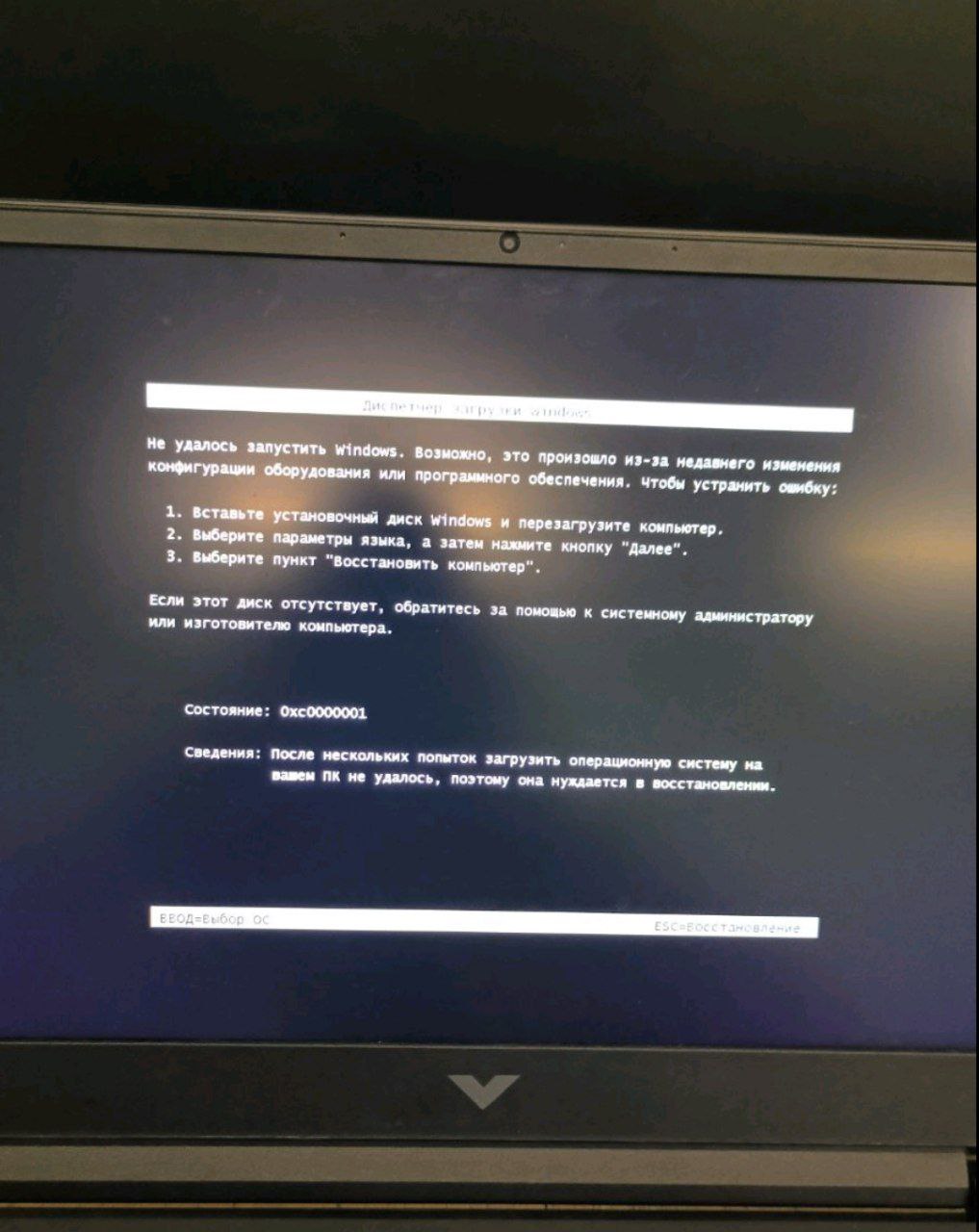
cout << "Усё запісана ;3\n";//!!!!

return 0;

}



**Прабачце, калі ласка, што не паспела да 00:00. Вось так атрымалася…**

****