МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования «Полоцкий государственный университет»

Факультет информационных технологий

Кафедра технологий программирования

**Отчёт по лабораторной работе № 3 по курсу «Программирование сетевых приложений»**

«Реализация линейной структуры данных «Очередь» и основные

алгоритмы обработки»

Вариант №11

ВЫПОЛНИЛ студент группы 21-ИТ-1

Шиковец Е.А.

ПРОВЕРИЛ преподаватель

Виноградова А.Д.

Полоцк, 2022 г.

**ЦЕЛЬ РАБОТЫ:** ознакомиться с основами линейной структуры данных «Очередь», изучить основные алгоритмы обработки ЛСД «Очередь», научиться применять полученные знания на практике.

**ТЕОРИТИЧЕСКИЕ СВЕДЕНИЯ**

1. Определение понятия очередь.

Очередью называется упорядоченный набор элементов, которые

могут удаляться с её начала и помещаться в её конец.

2. Опишите принцип работы очереди.

Очередь в программировании используется, как и в реальной жизни, когда нужно совершить какие-то действия в порядке их поступления, выполнив их последовательно.

3. Операции для работы с очередью.

Простейшие операции с очередью:

• init() инициализация очереди.

• insert (q, x) — помещение элемента x в конец очереди q (q —

указатель на очередь);

• x=remove (q) — удаление элемента x из очереди q;

• isempty(q) — возвращает 1, если очередь пуста и 0 в противном

случае;

• print(q) – вывод элементов очереди q.

4. Перечислите способы реализации очереди.

Существует несколько способов реализации очереди:

• с помощью одномерного массива;

• с помощью связанного списка;

• с помощью класса объектно-ориентированного программирования.

**ОПИСАНИЕ ПРОДЕЛАННОЙ РАБОТЫ:**

При выполнении работы я пользовался IDE Visual Studio Code. В результате выполнения работы был написан следующий код:

1 Часть:

#include <iostream>

using namespace std;

int main()

{

int tail = 0, head = 0, num, n;

int \*queue = new int;

cout << "Enter N: ";

cin >> n;

while (!n) // проверка на ввод

{

cin >> n;

}

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

{

cout << "Enter " << i + 1 << "st element of queue: ";

cin >> num;

queue[tail] = num;

tail++;

}

cout << "\n\nQueue before:\n";

for (int i = head; i < tail; i++) // вывод очереди до редактирования

{

cout << queue[i] << " ";

}

queue[head] = num; // удаление первого элемента очереди

head++;

queue[tail] = num; // удаление последнего элемента очереди

tail--;

cout << "\n\nQueue after:\n";

for (int i = head; i < tail; i++) // вывод очереди после редактирования

{

cout << queue[i] << " ";

}

}

2 часть:

#include <iostream>

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);

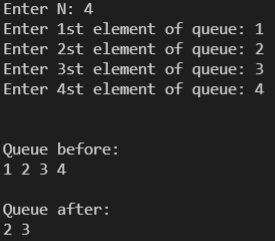
pop(&pq);

}

return 0;

}

**РЕЗУЛЬТАТЫ ВЫПОЛЕНИЯ ЛАБОРАТОРНОЙ РАБОТЫ**:



**Вывод:** в результате выполнения лабораторной работы была реализована однонаправленная очередь из n чисел, приоритетная очередь, было выполнено индивидуальное задание. Также я научился работать с очередями на ЯП C++.