

**МИНОБРНАУКИ РОССИИ**  
**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ**  
**ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**  
**«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)**  
**Кафедра МОЭВМ**

**отчет**  
**по лабораторной работе №3**  
**по дисциплине «Алгоритмы и структуры данных»**  
**Тема: Стеки и очереди**

Студент гр.8304

\_\_\_\_\_

Холковский К.В.

Преподаватель

\_\_\_\_\_

Фирсов М.А.

Санкт-Петербург

2019

### **Задание.**

Вариант :2-д (с реализацией очереди через динамическую (связанную) память (на базе указателей))

Содержимое заданного текстового файла *F*, разделенного на строки, переписать в текстовый файл *G*, перенося при этом в конец каждой строки все входящие в нее цифры (с сохранением исходного взаимного порядка как среди цифр, так и среди остальных литер строки).

### **Цель работы.**

Решить полученную задачу, используя очередь, реализованную на базе списка. Получения навыков работы с нелинейными структурами данных.

### **Описание алгоритма.**

Из файла *F* считывается строка, ведется проход по ней и если какой-то элемент является числом, то в очередь добавляется этот элемент. После прохода по этой строке в файл *G* выводится исходная строка и все содержимое очереди. Из файла *F* считывается ...

### **Описание функций и структур данных.**

- `Struct queue;` - хранит данные о верхнем элементе очереди и содержит методы для работы с очередью.
- `Struct Elem;` - хранит символ(цифру), указатель на следующий элемент и конструктор по умолчанию с деструктором.
- `queue::queue()` - конструктор по умолчанию, записывает в голову нулевой указатель.
- `queue::~~queue()` - деструктор, высвобождает память в голове.
- `Bool Queue::isEmpty()` - дает информацию о пустоте/заполненности очереди.
- `Void queue::add(char)` - добавляет в очередь новый элемент.
- `Char queue::pop()` - удаляет голову, и возвращает значение, которое в ней хранилось, смещая голову к следующему элементу.
- `Elem::Elem()` - конструктор по умолчанию, присваивает указателю на следующий элемент нулевой указатель.
- `Elem::~~Elem()` - деструктор, вызывает деструктор у следующего элемента.

## Выводы.

Для решения полученной задачи нелогично было использовать какую-либо из структур данных, т.к найденные данные можно было сразу выводить в файл.

## Тестирование

```
f8327yhf92fh1hn98hg20g294h29 - 8327921982029429
83fb923fh2983h2bg - 8392329832
g287hg4b39h298392bi34343u2bgier - 287439298392343432
3859238701957109571097511865198 - 3859238701957109571097511865198
18751097501957019571059710957 - 18751097501957019571059710957
185185701957150971 - 185185701957150971
((((((KJFIONO>>OPMOINIOjf0j320>>>MMM22<<<<<<< - 032022
1597105971509157105 - 1597105971509157105
fh982h29fh5353gn2gh209ghb29h2 - 9822953532209292
0hg209g20gh29j2gj - 020920292
g92j0902g9n209gn2 - 92090292092
n2g092g02hg02gh0294hg02 - 20920202029402
f2n8hg28h2424g0824h248 - 282824240824248
23h9g829g82h328hg2 - 239829823282
oiwhngwlegnowigqigbqoignq - No Digit heare
///eg/ge/ge/gegegegong0393g0m303g - 03930303
`n`i`nin`ir1/g2plgl2-k0gw[w[eg[e/?ekg29j904 - 122029904
wiog          nwiuti4gn          38h767g93h3gb3298h2 - 43876793332982
209rj238u2749284 - 2092382749284
19 - 19
- No Digit heare
1 - 1
a - No Digit heare
```

Пример вывода программы

<b>f8327yhf92fh1hn98hg20g294h29</b>	<b>8327921982029429</b>
<b>oiwhngwlegnowigqigbqoignq</b>	<b>No Digit heare</b>
	<b>No Digit heare</b>
<b>209rj238u2749284</b>	<b>2092382749284</b>

## Исходный код

```
#include <iostream>
#include <fstream>
```

```

struct queue
{
    struct Elem
    {
        Elem(char new_val):val(new_val),next(nullptr){}
        ~Elem(){
            delete next;
        }
        char val;
        Elem* next;
    };

    queue():Head(nullptr){}
    ~queue(){
        delete Head;
    }

    bool isEmpty(){
        if(Head == nullptr) return true;
        return false;
    }

    void add(char new_val){
        if(Head == nullptr)
            Head = new Elem(new_val);
        else
        {
            Elem* tmp = Head;
            while(tmp->next != nullptr)
                tmp = tmp->next;
            tmp->next = new Elem(new_val);
        }
    }

    char pop(){
        if(Head == nullptr)
        {
            std::cout << "Really?" << std::endl;
            return 0;
        }
        Elem* tmp = Head;
        char val_ = Head->val;
        Head = Head->next;
    }
};

```

```

        tmp->next = nullptr;
        delete tmp;
        return val_;
    }

private:
    Elem* Head;
};

int main(int argc, char* argv[]){
    if(argc==1)
        std::cout << "Need input filename" << std::endl;
    else
    {
        std::ifstream in(argv[1]);
        if (!in.is_open()){
            std::cout << "Can't open file" << std::endl;
            return 0;
        }
        if (in.eof())
        {
            std::cout << argv[1] << "Is empty File" << std::endl;
            return 0;
        }
        std::ofstream out;
        if(argc==3)
            out.open(argv[2]);
        else
            out.open("G");
        std::string cur_str;
        queue* point = new queue;
        while(std::getline(in, cur_str))
        {
            for(size_t i = 0; i < cur_str.size() ; i++)
                if(isdigit(cur_str[i]))
                    point->add(cur_str[i]);
            out << cur_str<< " - ";
            if(point->isEmpty())
                out << "No Digit heare";
            while(!point->isEmpty())
                out << point->pop();
            out << std::endl;
        }
        delete point;
    }
}

```

```
std::cout << "Work is Done" << std::endl;  
}
```