|  |  |  |
| --- | --- | --- |
|  | | |
| Федеральное государственное бюджетное  образовательное учреждение высшего образования «Новосибирский государственный технический университет» | | |
|  | | |
| Кафедра прикладной математики | | |
| Практическое задание № 2 | | |
| по дисциплине «Структуры данных и алгоритмы» | | |
| **Линейные списки** | | |
|  | | |
|  | Бригада 7 | Дроздов даниил |
| Группа ПМ-15 | иванова дарья |
| Вариант 4 | Чепурная Алина |
|  |  |
|  |  |
|  |  |
| Преподаватель | Тракимус Юрий Викторович |
|  |  |
| Новосибирск, 2022 | | |

1. **Условие задачи**

Описать динамическую очередь и написать операции работы с ней.

Вид очереди - линейный односвязный список двумя указателями.

К основным операциям добавить операцию, показывающую содержимое структуры после выполнения какого-либо действия с ней.

Эту операцию реализовать на основе базовых операций:

1. Очистить очередь
2. Поместить элемент в очередь
3. Взять элемент из очереди
4. Проверка очереди на пустоту
5. **Анализ задачи**

Входные данные: a ∈ N – номер операции над очередью, N ∈ Z – элемент очереди.

Выходные данные: “” – результат выполнения операции, N ∈ Z – элемент очереди.

Для решения задачи определяем структуру queue и структуру list. Структура queue представляет собой модель динамической очереди. Её элементы: beg – указатель на начало очереди, end - указатель на конец очереди. Сама очередь при этом состоит из элементов структуры list. Структура list представляет собой модель динамического списка со следующими элементами: elem – символьное значение элемента и next – указатель на следующий элемент списка. Структура queue также обладает методами для работы с динамической очередью:

1. empty - проверка очереди на пустоту.
2. clear - очистка очереди.
3. push - добавление нового элемента.
4. pop - выталкивание элемента.
5. print - вывод всех элементов очереди в консоль.

Для удобной работы с деком в функции main создаем консольное меню: выводим пронумерованный список методов (операций), добавляя операцию завершения программы. Затем запрашиваем номер операции, после, с помощью switch, запускаем запрошенную операцию.

1. **Программа**

#include <stdio.h>

#include <windows.h>

#include <conio.h>

enum command { PRINT = '0', CLEAR, PUSH, POP, EMPTY, EXIT };

struct list

{

char elem;

list \*next;

list(char \_elem = 0, list \*\_next = NULL) : elem(\_elem), next(\_next) {}

};

struct queue

{

list \*beg;

list \*end;

queue(list \*\_beg = NULL, list \*\_end = NULL) : beg(\_beg), end(\_end) {}

bool empty()

{

return !beg;

}

void clear()

{

for (char elem; pop(&elem); );

}

void push(char elem)

{

if (empty())

{

end = new list(elem);

beg = end;

} else

{

end->next = new list(elem);

end = end->next;

}

}

bool pop(char \*elem)

{

if (empty()) return false; // Очередь пуста

\*elem = beg->elem;

list \*d = beg;

beg = beg->next;

delete d;

return true;

}

void print()

{

queue q;

char i = 0;

for (; pop(&i); q.push(i))

printf\_s("%c ", i);

for (; q.pop(&i); push(i));

}

};

int main()

{

UINT inCp = GetConsoleCP(), outCp = GetConsoleOutputCP();

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

queue \*q = new queue;

bool repeat = true;

do

{

char cmd = 0, elem = 0;

printf\_s("\nВыберите команду: \n");

printf\_s("%c - Вывести очередь\n", PRINT);

printf\_s("%c - Очистить очередь\n", CLEAR);

printf\_s("%c - Добавить элемент в очередь\n", PUSH);

printf\_s("%c - Вытолкнуть элемент из очереди\n", POP);

printf\_s("%c - Проверить, пуста ли очередь\n", EMPTY);

printf\_s("%c - Завершить работу\n", EXIT);

printf\_s("Действие: ");

scanf\_s("\n%c", &cmd, 1);

switch (cmd)

{

case PRINT:

printf\_s("Содержимое очереди:\n");

q->print();

break;

case CLEAR:

printf\_s("Очередь очищена.\n");

q->clear();

break;

case PUSH:

printf\_s("Введите элемент: ");

scanf\_s("\n%c", &elem, 1);

q->push(elem);

break;

case POP:

q->pop(&elem) ? printf\_s("Полученный элемент: %c\n", elem) :

printf\_s("Не удалось получить элемент: очередь пуста.\n");

break;

case EMPTY:

q->empty() ? printf\_s("Очередь пуста.\n") :

printf\_s("Очередь не пуста.\n");

break;

case EXIT:

repeat = false;

break;

default:

break;

}

} while (repeat);

SetConsoleCP(inCp);

SetConsoleOutputCP(outCp);

return 0;

}

1. **Набор тестов**

|  |  |  |
| --- | --- | --- |
| **№** | **Входные данные** | **Назначение** |
| **1** | Действие: 2  Элемент: a  Действие: 2  Элемент: b  Действие: 2  Элемент: 3  Действие: 0  Действие: 4  Действие: 3  Действие: 0  Действие: 3  Действие: 1  Действие: 3  Действие: 1  Действие: 5 | Проверка на корректное выполнение программы. |

1. **Результаты работы программы**

|  |  |
| --- | --- |
| **№** | **Ввод/вывод программы** |
| **1** | Выберите команду:  0 - Вывести очередь  1 - Очистить очередь  2 - Добавить элемент в очередь  3 - Вытолкнуть элемент из очереди  4 - Проверить, пуста ли очередь  5 - Завершить работу  Действие: 2  Введите элемент: a  Действие: 2  Введите элемент: b  Действие: 2  Введите элемент: 3  Действие: 0  Содержимое очереди:  a b 3  Действие: 4  Очередь не пуста.  Действие: 3  Полученный элемент: a  Действие: 0  Содержимое очереди:  b 3  Действие: 3  Полученный элемент: b  Действие: 1  Очередь очищена.  Действие: 3  Не удалось получить элемент: очередь пуста.  Действие: 1  Очередь очищена.  Действие: 5 |