

**МИНОБРНАУКИ РОССИИ**

**федеральное государственное бюджетное образовательное учреждение**

**высшего образования**

**«Московский государственный технологический университет «СТАНКИН»**

**(ФГБОУ ВО «МГТУ «СТАНКИН»)**

|  |  |
| --- | --- |
| **Институт**  **информационных**  **технологий** | **Кафедра**  **информационных технологий и вычислительных систем** |

**КУРСОВАЯ РАБОТА**  
ПО ДИСЦИПЛИНЕ

«СТРУКТУРЫ И АЛГОРИТМЫ ОБРАБОТКИ ДАННЫХ»

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| СТУДЕНТА | *2* | КУРСА | | *бакалавриата* | ГРУППЫ | *ИДБ-20-03* |
|  | | | *(уровень профессионального образования)* | |  | |

|  |
| --- |
| **ГОРДЕЙЧИКА ДАНИИЛА АЛЕКСЕЕВИЧА** |
| *(ФИО)* |

ТЕМА РАБОТЫ

|  |
| --- |
| Множество последовательностей типа float |

|  |  |
| --- | --- |
| Направление: | 09.03.01 Информатика и вычислительная техника |
| Профиль подготовки: | Программное обеспечение средств вычислительной техники и автоматизированных систем |

|  |  |  |  |
| --- | --- | --- | --- |
| Отчет сдан «\_\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_20\_\_\_г. | | | |
|  |  |  |  |
| Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | | | |
|  | | | |
| Преподаватель | Лакунина О.Н., ст. преподаватель |  |  |
|  | *(Ф.И.О., должность, степень, звание.)* |  | *(подпись)* |

МОСКВА 2021

Оглавление

[Задание на курсовую работу 3](#_Toc59125181)

[Описание логических и физических структур данных 4](#_Toc59125182)

[Конечная схема реализуемой структуры](#_Toc59125183) 7

[Описание структур на языке С++](#_Toc59125185) 8

[Схема вызова функций](#_Toc59125186) 9

[Список функций и их назначение](#_Toc59125187) 10

[Исходный код программы с комментариями 18](#_Toc59125188)

# Задание на курсовую работу

Написать программу, реализующую логическую структуру данных – множество последовательностей типа float.

Программа должна работать в диалоговом режиме.

Каждая операция должна быть реализована в виде отдельной функции.

Множество и последовательность должны быть реализованы на базе структуры хранения – односвязный список.

Написать отчет по курсовой работе.

**Описание логических и физических структур данных**

**Множество** – логическая структура данных, представляющая из себя неупорядоченный набор уникальных элементов одного типа.

Элемент множества

Множество

Схема множества (на базе односвязного списка)

Н

null

С

Д

Д

С

Д

|  |  |
| --- | --- |
| Н | Указатель на начало односвязного списка |
| Д | Указатель на последовательность |
| С | Указатель на следующий элемент в односвязном списке |

**Список реализуемых функций**

1. Начать работу со множеством.
2. Закончить работу со множеством.
3. Сделать множество пустым.
4. Проверка: множество пусто / не пусто.
5. Удалить элемент из множества.
6. Взять элемент из множества
7. Добавить элемент во множество
8. Проверить принадлежит ли элемент множеству
9. Распечатать множество
10. Завершить работу программы

**Последовательность –** это линейно упорядоченный набор элементов, и элементы последовательности в каждый момент времени можно разделить на две части: прочитанную и непрочитанную. Для указания границы между прочитанной и непрочитанной частями служит указатель очередного элемента, который показывает на первый элемент из непрочитанной части.

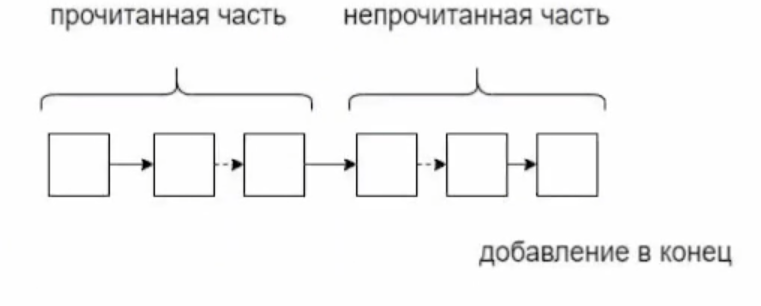
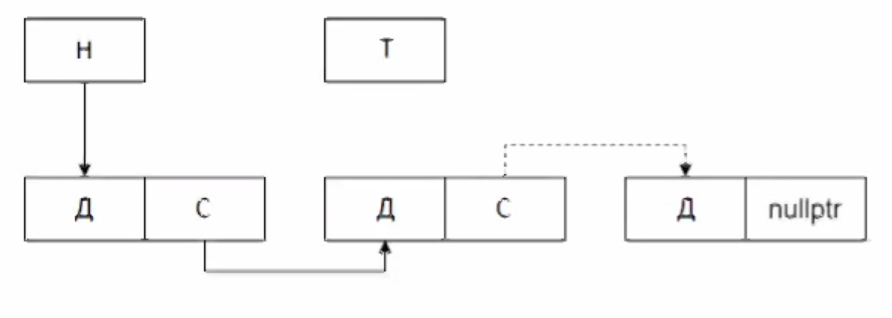


Схема последовательности (на базе односвязного списка)



|  |  |
| --- | --- |
| Н | Указатель на начало последовательности |
| Т | Указатель на текущий элемент |
| Д | Указатель на данные float |
| С | Указатель на следующий элемент |

**Список реализуемых функций:**

1. Начать работу с последовательностью.
2. Закончить работу с последовательностью.
3. Сделать последовательность пустой.
4. Проверка: последовательность пуста / не пуста.
5. Показать значение очередного элемента.
6. Пропустить очередной элемент
7. Прочесть очередной элемент
8. Изменить значение очередного элемента
9. Добавить элемент в конец
10. Установить указатель очередного элемента в начало
11. Проверка наличия непрочитанных элементов
12. Распечатать последовательность

**Конечная схема реализуемой структуры данных**

Последовательность (на базе односвязного списка)

Множество (на базе односвязного списка)

\*head

\*data

\*next

\*data

\*next

\*data

null

float data

\*next

float data

\*next

float data

null

\*head

\*pointer

**Описание структур на языке С++**

struct SetNode { //Узел множества

SequenceNode\* data; //Элемент данных множества

SetNode\* next;//Указатель на следующий элемент множества

};

struct SequenceNode { //Узел последовательности

float data; //Элемент данных последовательности

SequenceNode\* next;//Указатель на следующий элемент

//последовательности

};

**Схема вызова функций**

main()

SetMenu()

PrintSetMenu()

GetCommandofSet ()

CheckStart()

CheckCmdifSetisClear()

isSetClearandshowmsg()

GetStartedSet()

FinishWorkSet()

ClearSet()

isSetClear()

DeleteElement()

TakeElement()

AddtoSet()

isElementinSet()

PrintSet()

ClearSet()

SequenceMenu()

PrintSequence()

PrintSet()

PrintSequenceMenu()

GetCommandofSequence()

CheckcmdifitisClear()

isItClearandshowmsg()

GetStartedSequence()

FinishWorkSequence()

ClearMemory()

isItClear()

PrintElement()

SkipElement()

ReadElement()

ChangeElement()

AddElement()

SetPointerToHead()

isThereUnread()

InitializeSetElement()

isAllocated()

areSeqstheSame()

InitializeSequenceElement()

**Список функций и их назначение**

**Множество**

1. void SetMenu()

Функция-обработчик меню

1. void GetCommandofSet(SetNode\*& head, SetNode\*& object, bool& isStarted)

Обработка команды, которую ввел пользователь

1. void GetStartedSet(bool& isStarted)

Начать работу со множеством

1. void FinishWorkSet(SetNode\*& head, bool& isStarted)

Закончить работу со множеством

1. void ClearSet(SetNode\*& head)

Удалить множество

1. void isSetClear(SetNode\* head)

Проверка: множество пустое / не пустое

1. void DeleteElement(SetNode\*& head)

Удалить элемент множества

1. void TakeElement(SetNode\*& head, SetNode\*& object)

Взять элемент множества

1. void AddtoSet(SetNode\*& head)

Добавить элемент во множество

1. void isElementinSet(SetNode\* head)

Проверить наличие элемента во множестве

1. void PrintSet(SetNode\* head, int printMode)

Распечатать множество

1. bool CheckCommandifitisClear(int command)

Проверить команду на допустимость, если множество пустое

1. SequenceNode\* isItClearandShowMsg(SequenceNode\* head)

Проверить пусто ли множество и вывести сообщение

**Последовательность**

1. void SequenceMenu(SequenceNode\*& head)

Функция-обработчик меню

1. int GetCommandofSequence(SequenceNode\*& head, SequenceNode\*& pointer, SequenceNode\*& object, bool& isStarted)

Обработка команды, которую ввёл пользователь

1. void GetStartedSequence(bool& isStarted)

Начало работы с последовательностью

1. void FinishWorkSequence(SequenceNode\*& head)

Закончить работу с последовательностью

1. void ClearMemory(SequenceNode\*& head, SequenceNode\*& pointer)

Очистить последовательность

1. void isItClear(SequenceNode\* head, SequenceNode\* pointer)

Проверка: последовательность пуста / не пуста

1. void PrintElement(SequenceNode\* pointer)

Вывести на экран очередной элемент

1. void SkipElement(SequenceNode\*& pointer)

Пропустить очередной элемент

1. void ReadElement(SequenceNode\*& pointer, SequenceNode\*& object)

Прочитать очередной элемент

1. void ChangeElement(SequenceNode\*& pointer)

Изменить элемент

1. void AddElement(SequenceNode\*& head, SequenceNode\*& pointer)

Добавить элемент

1. void SetPointerToHead(SequenceNode\* head, SequenceNode\*& pointer)

Установить указатель в начало

1. void isThereUnread(SequenceNode\* pointer)

Проверить есть ли непрочитанные элементы

1. void PrintSequence(SequenceNode\* head, SequenceNode\* pointer)\

Распечатать последовательность

**Исходный код программы с комментариями**

**Файл main.cpp**