Отчет по лабораторной работе № 26 по курсу Фундаментальная информатика

Студент группы М8О-204Б-22, Филиппов Фёдор Иванович, № по списку 18

Контакты: gooselinjk@yandex.ru

Работа выполнена: “6” октября 2023 года

Преподаватель: Потенко М.А., каф.806

Входной контроль знаний с оценкой_____

Отчёт сдан “7” октября 2023 года, ИО ____

Подпись преподавателя _____

1. Тема: Абстрактные типы данных. Рекурсия. Модульное программирование на языке С. Автоматизация сборки программ модульной структуры на языке С с использованием утилиты make.

2. Цель работы: Составить и отладить модуль определений и реализации абстрактного типа данных – стек. Составить Makefile - файл, содержащий набор инструкций для утилиты make по запуску программы.

3. Задание *(вариант № S1)***:** Структура стек. Процедура поиска и удаления максимального элемента методом сортировки линейного выбора

4. Оборудование

ЭВМ — ноутбук HP, процессор — Ryzen 5500U, с ОП 16384 МБ и НМД 1048576 МБ, Терминал Windows Powershell (с возможностью переключения на UNIX)

5. Программное обеспечение

Операционная система семейства Windows, наименование Windows 11 Home, версия 22H2

Редактор текстов — Sublime Text

Утилиты операционной системы — терминал Windows Powershell

Прикладные системы и программы — Visual Studio Code, Visual Studio

6. Идея, метод, алгоритм решения задачи (в формах: словесной, псевдокода, графической или формальные с пред- и постусловиями)

Файл "mystack.h" объявляет структуры и функции, необходимые для работы с стеком. В нем определены структуры StackNode и Stack, которые представляют собой узел стека и сам стек соответственно. Также в файле содержатся прототипы функций, такие как create_node, create_stack, is_empty, push, pop, check_top, delete_stack, print_stack, delete_max_elem и sort_stack. Этот файл описывает структуры данных и основные операции, которые можно выполнять с ними.

Файл "mystack.c" содержит реализацию функций, объявленных в "mystack.h". В нем находятся реализации функций для создания узла, создания стека, проверки на пустоту, добавления элемента в стек, извлечения элемента из стека, проверки верхнего элемента, удаления стека, вывода стека на экран, а также функции для удаления максимального элемента и сортировки стека. Этот файл содержит конкретную реализацию структур данных и их операций.

Файл "main.c" представляет собой главный исходный файл программы. В нем реализовано взаимодействие с пользователем через командный интерфейс. Программа предлагает пользователю различные действия, такие как добавление элемента в стек, извлечение элемента, вывод стека на экран, сортировку стека и удаление максимального элемента. В зависимости от выбора пользователя выполняются соответствующие функции из "mystack.c". Этот файл является точкой входа в программу и обеспечивает взаимодействие с пользователем.

Метод сортировки, используемый в функции sort_stack() данной программы, является методом сортировки линейным выбором (Selection Sort). Он работает следующим образом:

Сначала проверяется, не пуст ли стек. Если стек пуст, то выводится сообщение о том, что стек пуст, и сортировка завершается.

Создается новый временный стек sorted_stack, который будет содержать элементы в упорядоченном порядке.

В основном стеке начинается процесс сортировки. Для этого:

Извлекается верхний элемент top из основного стека с помощью функции pop().

Перебираются все оставшиеся элементы в основном стеке и сравниваются с top.

Если находится элемент, который меньше top, то этот элемент извлекается из основного стека и помещается в sorted_stack.

Если находится элемент, который больше или равен top, он остается в основном стеке.

Этот процесс повторяется до тех пор, пока основной стек не опустеет.

После завершения этого процесса, все элементы из основного стека, начиная с самых маленьких, были извлечены и помещены в sorted_stack, что приводит к тому, что sorted_stack содержит элементы изначального стека в упорядоченном порядке.

Затем элементы из sorted_stack обратно помещаются в основной стек, в результате чего основной стек теперь содержит элементы в порядке возрастания.

Временный стек sorted_stack освобождается, так как его задача выполнена.

7. Сценарий выполнения работы (план работы, первоначальный текст программы в черновике и тесты, либо соображения по тестам)

Когда я начал разработку данной программы для сортировки стека, мой первый шаг был определить, какой метод сортировки стека я хотел бы использовать. Поскольку стек - это структура данных с ограниченным доступом, то мне пришла идея использовать метод сортировки выбором, так как он позволяет на каждом шаге находить максимальный элемент и перемещать его в отсортированный стек.

С планом реализации я решил начать с создания необходимых структур данных и функций. Начал с определения структуры `StackNode` для узлов стека и структуры `Stack` для самого стека. Затем создал функции для создания стека (`create_stack`), создания узла (`create_node`), проверки стека на пустоту (`is_empty`), добавления элемента в стек (`push`), удаления элемента из стека (`pop`) и проверки значения вершины стека без его удаления (`check_top`).

После того как базовые структуры и функции были реализованы, перешел к разработке функций, связанных с сортировкой стека. Создал функцию `delete_max_elem`, которая находит и удаляет максимальный элемент из стека, и функцию `sort_stack`, которая сортирует стек с использованием метода линейного выбора.

Чтобы убедиться в правильной работе программы, написал несколько тестовых случаев. Один из тестов проверял сортировку стека с произвольными числами, включая отрицательные и нули. Другой тест включал попытку сортировки пустого стека, чтобы убедиться, что программа корректно обрабатывает такой случай. Также проверил сортировку стека, содержащего один элемент, и удостоверился, что стек остается неизменным в этом случае. По завершении тестов программа успешно сортировала стек и прошла все тестовые сценарии.

8. Распечатка протокола (подклеить листинг окончательного варианта программы с тестовыми примерами)

```
1. Push (add)
2. Pop (extract)
3. Print stack
4. Sort stack
5. Delete largest element
6. Exit
1
Enter a value: 5
Element is added.
1. Push (add)
2. Pop (extract)
3. Print stack
4. Sort stack
5. Delete largest element
6. Exit
3
Stack:
[ 5 ]
===
1. Push (add)
2. Pop (extract)
3. Print stack
4. Sort stack
5. Delete largest element
6. Exit
1
Enter a value: 2
Element is added.
1. Push (add)
2. Pop (extract)
3. Print stack
4. Sort stack
5. Delete largest element
6. Exit
3
Stack:
[ 2 ]
[ 5 ]
===
```

```
===
1. Push (add)
2. Pop (extract)
3. Print stack
4. Sort stack
5. Delete largest element
6. Exit
4
Stack is sorted
1. Push (add)
2. Pop (extract)
3. Print stack
4. Sort stack
5. Delete largest element
6. Exit
4
Stack is sorted
1. Push (add)
2. Pop (extract)
3. Print stack
4. Sort stack
5. Delete largest element
6. Exit
3
Stack:
[ 5 ]
[ 2 ]
===
1. Push (add)
2. Pop (extract)
3. Print stack
4. Sort stack
5. Delete largest element
6. Exit
6
PS C:\Users\theo_rvn\Desktop\coding\26>
```

```
Stack:
[ 6 ]
[ 2 ]
[ 1 ]
[ 83 ]
[ 56 ]
[ 21 ]
[ 23 ]
===
1. Push (add)
2. Pop (extract)
3. Print stack
4. Sort stack
5. Delete largest element
6. Exit
2
Highest element [ 6 ] extracted.
1. Push (add)
2. Pop (extract)
3. Print stack
4. Sort stack
5. Delete largest element
6. Exit
3
Stack:
[ 2 ]
[ 1 ]
[ 83 ]
[ 56 ]
[ 21 ]
[ 23 ]
===
1. Push (add)
2. Pop (extract)
3. Print stack
4. Sort stack
5. Delete largest element
6. Exit
4
Stack is sorted
1. Push (add)
2. Pop (extract)
3. Print stack
4. Sort stack
5. Delete largest element
6. Exit
3
```

```
Stack is sorted
1. Push (add)
2. Pop (extract)
3. Print stack
4. Sort stack
5. Delete largest element
6. Exit
3
Stack:
[ 83 ]
[ 56 ]
[ 23 ]
[ 21 ]
[ 2 ]
[ 1 ]
===
```

9. **Дневник отладки** должен содержать дату и время сеансов отладки и основные события (ошибки в сценарии и программе, нестандартные ситуации) и краткие комментарии к ним. В дневнике отладки приводятся сведения об использовании других ЭВМ, существенном участии преподавателя и других лиц в написании и отладке программы.

№	Лаб. или дом.	Дата	Время	Событие	Действие по исправлению	Примечание

10. Замечания автора по существу работы

_____

_____

_____

_____

Недочеты при выполнении работы могут быть устранены следующим образом:_____

_____

11. Выводы: Метод сортировки выбором не является самым эффективным способом сортировки, особенно для больших стеков. Однако, он предоставляет хороший обучающий опыт и понимание работы алгоритмов сортировки.

В целом, выполнение этой лабораторной работы позволило мне лучше понять работу со структурой данных стек и применение алгоритмов сортировки к таким структурам. Это также подчеркнуло важность правильной работы с памятью и тщательное тестирование для обеспечения надежности программы.