|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **Министерство образования и науки Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ Информатика и системы управления (ИУ)

КАФЕДРА Программное обеспечение ЭВМ и информационные технологии (ИУ7)

**ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №4  
«Работа со стеком»**

Студент Фам Минь Хиеу

Группа ИУ7И – 32Б

Преподаватель Никульшина Т.А

*2022 г.*

1. **Условие задачи:**

Разработать программу работы со стеком, реализующую операции добавления и удаления элементов из стека и отображения текущего состояния стека. Реализовать стек: а) массивом; б) списком.

Ввести целые числа в 2 стека. Используя третий стек отсортировать все введенные данные.

1. **Описание ТЗ:**
2. **Входные и выходные данные:**
3. **Входные данные:**

Номер пункта от 0 до 6. Элементы стека - числа

1. **Выходные данные:**

Результат сортировки всех введенных данных , отображение текущего состояния стека (если это стек в виде списка, то также отображаются свободные области), сравнение алгоритмов.

1. **Функции программы:**
2. Ввести целые числа в 2 стека – либо в виде массива либо в виде списки
3. Вывести элементы списков. Выводятся адреса элементов стека, а также свободные области.
4. Отсортировать все введенные данные, при этом используется третий стек.
5. Удалить последний элемент стека - удаление элемента с вершины стека.
6. Добавить элемент в конец стека - добавление элемента на вершину стека.
7. Сравнительный анализ алгоритмов.
8. Выход - выход из программы.
9. **Способы обращения к программе:** Программа запускается через ./app.exe.
10. **Описание внутренних структур данных**

Структура для описания стека-массива:

***typedef struct***

***{***

***int a[MAX\_SIZE];***

***int \*ptr;***

***} arr\_stack\_t*;**

Где MAX\_SIZE - максимальный размер стека-массива;

a[MAX\_SIZE] - массив, содержащий значения элементов стека-массива;

ptr - указатель стека-массива.

Структура для описания стека-списка:

***typedef struct list\_stack***

***{***

***int data;***

***struct list\_stack \*prev;***

***} list\_stack\_t;***

Где data - значение элемента узла стека-списка;

prev - указатель на узел, который находится ниже в стеке-списке;

Структура для описания свободных областей:

***typedef struct***

***{***

***list\_stack\_t\* a[MAX\_SIZE];***

***int count;***

***} storage;***

Где a[MAX\_SIZE] -массив указателей на list\_stack\_t;

count – количество освобождаемых элементов стека.

1. **Описание алгоритма:**

Сортировка двух стеков реализуется на основе сортировки вставками с использованием третего стека такого же типа.

Добавление элемента для стека-списка реализуется выделением памяти под новый элемент и связыванием его с концом списка. Для стека-массива добавление элемента реализуется сдвигом указателя стека на одну позицию вперед, а затем записью очередного значения добавляемого элемента.

Удаление элемента для стека-списка реализуется переходом к предыдущему элементу и освобождением памяти из-под удаляемого элемента. Для стека-массива, удаление элемента - сдвиг указателя стека на одну позицию назад.0

1. **Тестирование программы:**

|  |  |
| --- | --- |
| ***Действие*** | ***Результат*** |
| Неверный ввод команды:  -1 | Сообщение: “Ошибка: команда должна быть в [0, 6] .” |
| Неверный ввод элемента стека:  Пункт 1  Подпункт 1  Ввод abc | Сообщение: “Ошибка ввода элемента стека: должно быть целым числом.” |
| Неверный выбор стека:  Пункт 1  Пункт 3 | Сообщение: “ Следуйте инструкциям в меню.” |
| Добавление элемента в стек, когда он достиг максимального размера | Сообщение:”Переполнение третего стека.” |
| Удаление элемента из пустого стека | Сообщение: ”стек пуст”. |
| Ошибка выделения памяти во время добавления элемента в стек-список | Сообщение: ”ошибка выделения памяти.” |
| Пункт 1.1  Ввод 1 4 2 3  Ввод 2  Пункт 2 | Дата была успешно введена. |
| Пункт 1.2  Ввод 1 4 2  Ввод -1 3 0  Пункт 2 |  |
| Пункт 1.1  Ввод 1 4 2 3  Ввод 2  Пункт 3 |  |
| Пункт 1.2  Ввод 1 4 2  Ввод -1 3 0  Пункт 3 |  |
| Пункт 1.1  Ввод 1 4 2 3  Ввод 2  Пункт 4.1  Пункт 4 |  |
| Пункт 1.2  Ввод 1 4 2  Ввод -1 3 0  Пункт 4  Пункт 2 |  |
| Пункт 1.1  Ввод 1 4 2 3  Ввод 2  Пункт 4.1  Пункт 5.1  Ввод 7 |  |
| Пункт 1.2  Ввод 1 2  Ввод 3 2  Пункт 5.1  Ввод 3  Пункт 2 |  |

1. **Оценка эффективности.**

*Сравнительный анализ:*



***Выводы:***

Практически эффективнее использовать стек в виде массива, особенно статического. Он выигрывает по времени с большим повторением и с большим количесвом элементов. Это происходит за счет того, что для статического массива память в процессе компиляции выделено заранее, а для списка по мере необходимости. Список выигрывает по памяти при малой заполненности массива (<50%) так как при работе с массивом выделяется память, но не используется.

1. **Ответы на контрольные вопросы:**
2. **Что такое стек?**

Стек - структура данных, характеризующаяся тем, что во время работы с ним доступен только последний элемент. Стек имеет последовательный доступ, т.е. нельзя просмотреть все элементы стека, не просмотрев каждый элемент последовательно, начиная с конца. Стек функционирует по правилу LIFO (last in, first out) - последним пришел - первым ушел.

1. **Каким образом и сколько памяти выделяется под хранение стека при различной его реализации?**

При реализации стека в виде массива используется статический массив, размер которого задается константой, таким образом размер стека-массива становится известным уже на момент компиляции (память выделяется на стеке). Также выделяется память под один указатель такого же типа, как и данные массива.

Конкретно выделяется по MAX\_SIZE \* sizeof(тип данных элемента) + sizeof(указатель на тот же тип данных), где MAX\_SIZE - максимальный размер стека-массива.

При реализации стека в виде списка память выделяется на куче при каждом добавлении элемента. Занимаемая память больше памяти для стека-массива, т.к. хранится и указатель на предыдущий элемент.

Для конкретной реализации выделяется по sizeof(узел стека), который включает в себя одно целое число, указатель на предыдущую структуру и указатель на максимальный адрес памяти для стека-списка.

1. **Каким образом освобождается память при удалении элемента стека при различной реализации стека?**

При удалении элемента из стека-списка освобождается память из-под верхнего узла, предварительно выполнив смещение к предыдущему узлу.

При удалении элемента из стека-массива происходит лишь смещение указателя стека на одну позицию назад. Память освобождается только по окончании работы с массивом.

1. **Что происходит с элементами стека при его просмотре?**

По мере просмотра стека элементы из него удаляются в соответствии с классической реализацией. Поэтому для реализации функции печати был использован дополнительный стек.

1. **Каким образом эффективнее реализовывать стек? От чего это зависит?**

Реализовывать стек при помощи списка эффективнее тем, что память для него ограничена размером оперативной памяти, в то время как для статического массива память ограничена размером стека. По времени работы реализация стека при помощи массива эффективнее