|  |  |
| --- | --- |
| 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 Ву Минь Куанг

Группа ИУ7И – 34Б

Преподаватель Силантьева А. В.

*2021 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:**

Целое число от 1 до 7, представляющее собой пункт меню программы. В зависимости от выбора пункта меню может потребоваться ввести дополнительную информацию, такую как значение добавляемого элемента или тип массива.

1. **Выходные данные:**

Командно-зависимо: печать результата сортировки всех введенных данных на экран, отображение текущего состояния стека (если это стек в виде списка, то также отображаются свободные области), сообщения о результате выполнения доступных операций, печать сравнительного анализа алгоритмов.

1. **Задачи, реализуемые программой:**

Программа способна решать следующие задачи:

1. Ввести целые числа в 2 стека - пользователь должен ввести целые числа, которые должны храниться в стеке, и после этого нажать клавишу Enter для завершения ввода, при этом необходимо выбрать тип стеков, в которых будет храниться информация. Это осуществляется посредством меню:
2. В виде массива
3. В виде списка
4. Вывести элементы списков - на экран выводится содержимое стеков. Для стека-списка дополнительно выводятся адреса элементов стека, а также свободные области (освобожденные ячейки памяти для наблюдения фрагментации или ее отсутствия).
5. Отсортировать все введенные данные - происходит сортировка двух стеков, при этом результат объединяется в один результирующий стек.

*Замечание: при сортировке содержимое двух исходных стеков очищается, таким образом их печать невозможна после выполнения этого пункта.*

1. Удалить последний элемент стека - удаление элемента с вершины стека. При этом необходимо выбрать, из какого стека удалить элемент. Это осуществляется посредством меню:
2. Первый
3. Второй
4. Добавить элемент в конец стека - добавление элемента на вершину стека. При этом необходимо ввести значение элемента после соответствующего сообщения и затем выбрать массив. Выбор массива осуществляется посредством меню:
5. Первый
6. Второй
7. Сравнительный анализ алгоритмов - печать сравнительного анализа операций исключения, добавления и сортировки для стека-массива и стека-списка на экран в количественном и процентном соотношении.
8. Справка - печать информации о программе на экран.
9. Выход - выход из программы.
10. **Способы обращения к программе:** пользователь работает с программой посредством меню, команды которого описаны выше.
11. **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 нижестоящий узел в стеке-списке;

Структура для описания свободных областей:

***typedef struct***

***{***

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

***int index;***

***} freed\_t;***

Где a[MAX\_SIZE] - аналогичен соотв. массиву для стека-массива;

index – количество освобождаемых элементов стека.

1. **Описание алгоритма:**

Сортировка двух стеков реализуется на основе сортировки вставками. Используется третий результирующий стек такого же типа, как и сортируемые стеки. В начале работы он пуст. По мере просмотра каждого из стеков снимается очередной элемент и затем его значение рассматривается относительно уже упорядоченных значений стека (на основе сортировки вставками). В конце сортировки исходные стеки становятся пустыми.

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

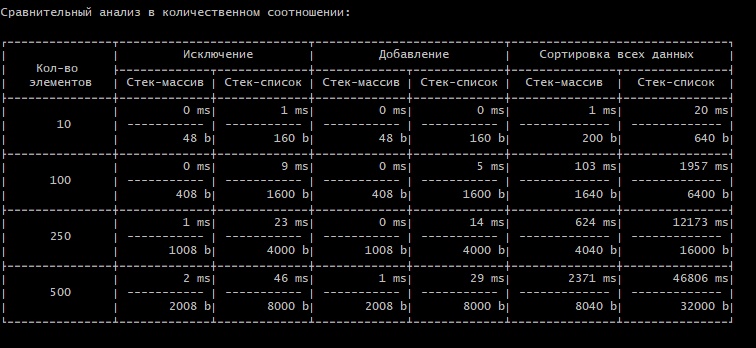
Добавление элемента для стека-списка реализуется выделением памяти под новый элемент и связыванием его с концом списка. Для стека-массива добавление элемента реализуется сдвигом указателя стека на одну позицию вперед, а затем записью очередного значения добавляемого элемента.

1. **Тестирование программы:**

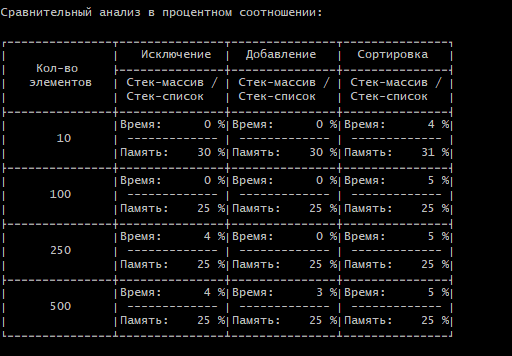
|  |  |
| --- | --- |
| ***Действие*** | ***Реакция программы*** |
| Неверный ввод команды:  -12 | Сообщение: “Ошибка: команда должна быть одним из чисел, предложенных в меню.”  Завершение работы программы. |
| Неверный ввод элемента стека:  Пункт 1  Подпункт 1  Ввод ук12 | Сообщение: “Ошибка ввода элемента стека: должно быть введено целое число.”  Завершение работы программы. |
| Неверный выбор стека:  Пункт 1  Пункт 3 | Сообщение: “Ошибка: команда должна быть одним из чисел, предложенных в меню.”  Завершение работы программы. |
| Добавление элемента в стек, когда он достиг максимального размера | Сообщение:” Ошибка: объединение двух стеков приведет к переполнению результирующего стека.”  Завершение работы программы. |
| Исключение элемента из пустого стека | Сообщение:” Невозможно произвести удаление: стек пуст”.  Завершение работы программы. |
| Ошибка выделения памяти во время добавления элемента в стек-список | Сообщение:” Невозможно добавить элемент в стек: ошибка выделения памяти.”  Завершение работы программы: |
| Пункт 1.1  Ввод 1 4 2 3  Ввод 2  Пункт 2 | Сообщение об успешности ввода данных. |
| Пункт 1.2  Ввод 1 4 2  Ввод -1 3 0  Пункт 2 | Сообщение об успешности ввода данных |
| Пункт 1.1  Ввод 1 4 2 3  Ввод 2  Пункт 3 | Сообщение об успешности ввода данных. |
| Пункт 1.2  Ввод 1 4 2  Ввод -1 3 0  Пункт 3 | Сообщение об успешности ввода данных. |
| Пункт 1.1  Ввод 1 4 2 3  Ввод 2  Пункт 4.1  Пункт 4 | Сообщение об успешности удаления элемента. |
| Пункт 1.2  Ввод 1 4 2  Ввод -1 3 0  Пункт 4  Пункт 2 | Сообщение об успешности удаления элемента. |
| Пункт 1.1  Ввод 1 4 2 3  Ввод 2  Пункт 4.1  Пункт 5.1  Ввод 7 | Сообщение об успешности добавления элемента. |
| Пункт 1.2  Ввод 1 2  Ввод 3 2  Пункт 5.1  Ввод 3  Пункт 2 | Сообщение об успешности добавления элемента. |

1. **Оценка эффективности.**

*Сравнительный анализ в количественном соотношении:*



*Сравнительный анализ в процентном соотношении:*



Для каждого измерения добавления и удаления проводилось 10000 опытов. Для измерения сортировки проводилось по 300 опытов.

***Выводы:***

Практически всегда эффективнее использовать стек в виде массива, особенно статического. Он выигрывает по всем параметрам с большим отрывом (в 5-6 раз по памяти и в 5-8 раз по времени в случае статической реализации). Это происходит за счет того, что для статического массива память выделяется на стеке, а для списка на куче. Доступ к стеку происходит гораздо быстрее, чем к куче.

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 заводится 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.

Для конкретной реализации 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.

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 промежуточный стек.

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

Практически всегда эффективнее использовать стек в виде массива, особенно статического. Он выигрывает по всем параметрам с большим отрывом (в 4 раза по памяти и больше 10 раз по времени в случае статической реализации). Однако память для хранения стека-массива ограничена размером стека, а память стека-списка ограничена лишь размером доступной оперативной памяти, т.к. память выделяется в куче, поэтому, когда размер стека достигает больших размеров, необходимо использовать стек в виде списка.