**Типы и Структуры Данных**

**Oтчет**

## *Работа № 4:* Работа со стеком

***Студент: Нгуен Фыок Санг***

***Группa: ИУ7И-36Б***

*2019*

1. **Oписание условия задачи:**

**Цель работы:** реализовать операции работы со стеком, который представлен в виде массива (статического или динамического) и в виде односвязного линейного списка; оценить преимущества и недостатки каждой реализации: получить представление о механизмах выделения и освобождения памяти при работе со стеком.

Создать программу работы со стеком, выполняющую операции добавления, удаления элементов и вывод текущего состояния стека. Реализовать стек: а) массивом; б) списком. Все стандартные операции со стеком должны быть оформлены подпрограммами. При реализации стека списком в вывод текущего состояния стека добавить просмотр адресов элементов стека и создать свой список или массив свободных областей (адресов освобождаемых элементов) с выводом его на экран.

Элементами стека являются слова. Распечатайте слова в обратном порядке.

**II. Oписание ТЗ:**

1. **Исходные данные**: строка со словами
2. **Результат**: Строкa со словами в обратном порядке
3. **Задачи, реализуемой программой**:
   * Распечатать stack в обратном порядке.

**III. Описание СД:**

1. **Для ввода:** использовать связанный списoк
2. **Для обработки:** использовать связанный список для сохранения адреса
3. **Для результатa:** использовать связанные списки

***структура узла:***

**struct node**

**{**

**char value[MAX\_LEN + 1];**

**struct node \* next;**

**}**

***структура стека***:

**struct stack\_1**

**{**

**node\_t head;**

**int n;**

**};**

**IV. Oписание алгоритма:**

1. ***Алгоритм вставки (push) node (P)***
   1. Если стек **заполнен**, появляется сообщение об ошибке
   2. Если стек **пуст**, поставить узел (P) как head
   3. P^next🡨 HEAD;
   4. HEAD🡨 P;
2. Aлгоритм отключения (pop)
   1. Если стек **пуст**, появляется сообщение об ошибке
   2. P 🡨 HEAD;
   3. HEAD 🡨 HEAD^next;
   4. P^next 🡨 Nil;
   5. Возвращает узел (P)

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| N = | 100 | 300 | 500 | 700 | 900 | 1000 | 2000 | 3000 | 4000 | 5000 | 7000 | 8000 | 9000 | 10000 |
| Time1 | 1 | 2 | 4 | 9 | 7 | 11 | 18 | 33 | 38 | 48 | 71 | 79 | 90 | 100 |
| Time2 | 1 | 5 | 10 | 15 | 22 | 22 | 82 | 177 | 233 | 348 | 606 | 999 | 1134 | 1307 |
| Aver1 | 0.01 | 0.007 | 0.008 | 0.013 | 0.008 | 0.011 | 0.009 | 0.011 | 0.009 | 0.01 | 0.01 | 0.01 | 0.01 | 0.01 |
| Aver2 | 0.01 | 0.017 | 0.02 | 0.021 | 0.024 | 0.022 | 0.041 | 0.059 | 0.058 | 0.07 | 0.087 | 0.125 | 0.126 | 0.131 |

**V. Набор тестов:**

\*Time1: Время для списка \*Time2: Время для массива

**VI. Ответы на вопросы:**

* ***Что такое стек***?
  + **Стек** – это последовательный список с переменной длиной, в котором включение и исключение элементов происходит только с одной стороны – с его вершины. Стек функционирует по принципу: последним пришел – первым ушел, ***Last In – First Out (LIFO).***
* ***Каким образом и сколько памяти выделяется под хранение стека при различной его реализации?*** 
  + Можно определить стек какнекий абстрактный тип данных (АТД). Преимущество использования АТД заключается в том, что в этом случае описание данных и операций, выполняемых над ними, отделено от их представления и реализации.
  + **Стек в виде массива**

размер массива = n \* (размер элемента)

* + **Стек в виде списка** ( n - количество узлов

размер стека = (размер узлa) \* n.

* + Размер узла больше, чем размер элемента массива, поэтому память, используемая для хранения списков, больше, чем память для хранения массивов.
* ***Каким образом освобождается память при удалении элемента стека при различной реализации стека?*** 
  + Стек в виде списка
    - Освободить память для содержимого узла (если применимо).
    - Освободить память для узла
  + Стек в виде массива
    - изменить размер (resize) массива: n 🡨 n – 1
* ***Каким образом эффективнее реализовывать стек? От чего это зависит?***
  + Сохранить стек в виде списка
    - Узлы находятся в разных областях памяти
    - Удалить и вставить быстрее
  + Сохранить стек в виде массива
    - Простая реализация
    - меньше памяти и времени