

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

1. Описание задания

В рамках задания необходимо:

1. Использовать:

- вычислительную систему с архитектурой x86-64;
- операционную систему Linux;
- языки программирования C/C++;
- более простые библиотеки уровня ОС и языка C: `stdio.h`, `stdlib.h`, `string.h` и т. д.

2. Разработать программу в виде консольного приложения, ориентированную на процедурный подход с использованием статически типизированного универсального языка программирования.

Запуск программы должен осуществляться из командной строки, в которой указываются: имя запускаемой программы; имя файла с исходными данными; имя файла с выходными данными.

Примеры:

`command -f infile.txt outfile1.txt outfile2.txt` (для ввода из файла)

`command -n number_of_elements outfile1.txt outfile2.txt` (для случайной генерации)

3. Реализовать в программе следующий функционал (**вариант 13**):

Обобщённый артефакт	Базовые альтернативы (и их отличительные параметры)	Общая для всех альтернатив переменная	Общая для всех альтернатив функция
Растение	1. Деревья (возраст – длинное целое) 2. Кустарники (месяц цветения)	Название – строка символов (макс. длина = 20 символов)	Частое от деления числа гласных букв в названии на

	– перечислимый тип) 3. Цветы (домашние, садовые, дикие – перечислимый тип)		общую длину названия
--	---	--	----------------------

4. Поместить данные объекты в контейнер и в соответствии с вариантом задания (**вариант 19**) удалить из контейнера те элементы, для которых значение, полученное с использованием функции, общей для всех альтернатив, меньше чем среднее арифметическое для всех элементов контейнера, полученное с использованием той же функции.

5. Провести отладку и тестирование разработанной программы на заранее подготовленных тестовых наборах данных (не менее 5). Тестовые данные с большим числом элементов должны порождаться программой с использованием генераторов случайных наборов данных. Управление вводом данных задается из командной строки.

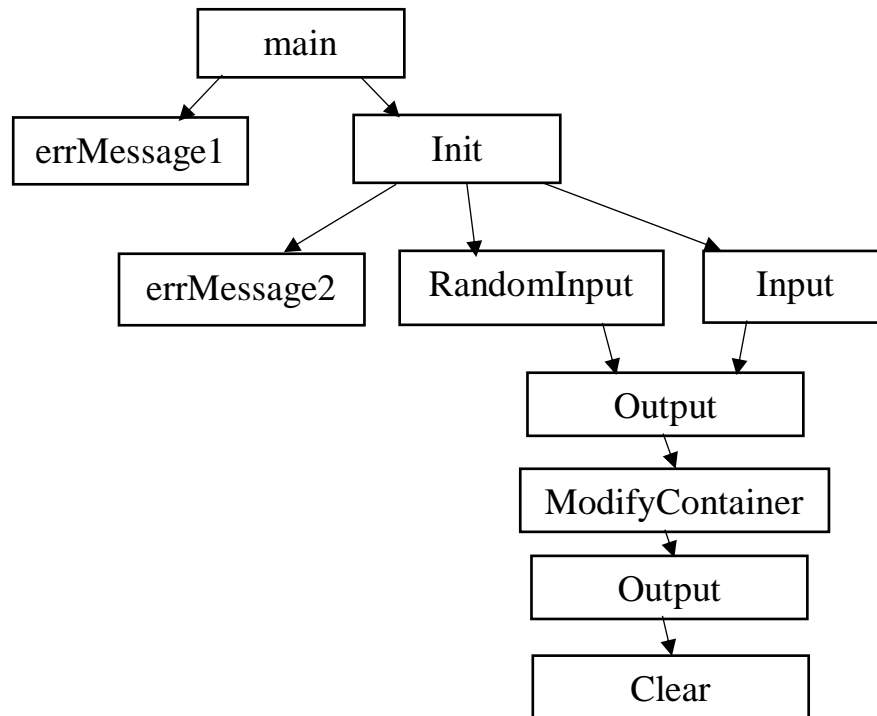
6. Создать отчёт по выполненному заданию, описав структуру используемой ВС с наложением на нее обобщённой схемы разработанной программы и зафиксировав основные характеристики программы.

2. Схема архитектуры ВС с размещённой на неё разработанной программой.

1. Схема памяти функции main.

Память программы
<pre>int main(int argc, char* argv[]) { ... }</pre>

Stack



Переменные программы (размер в байтах)

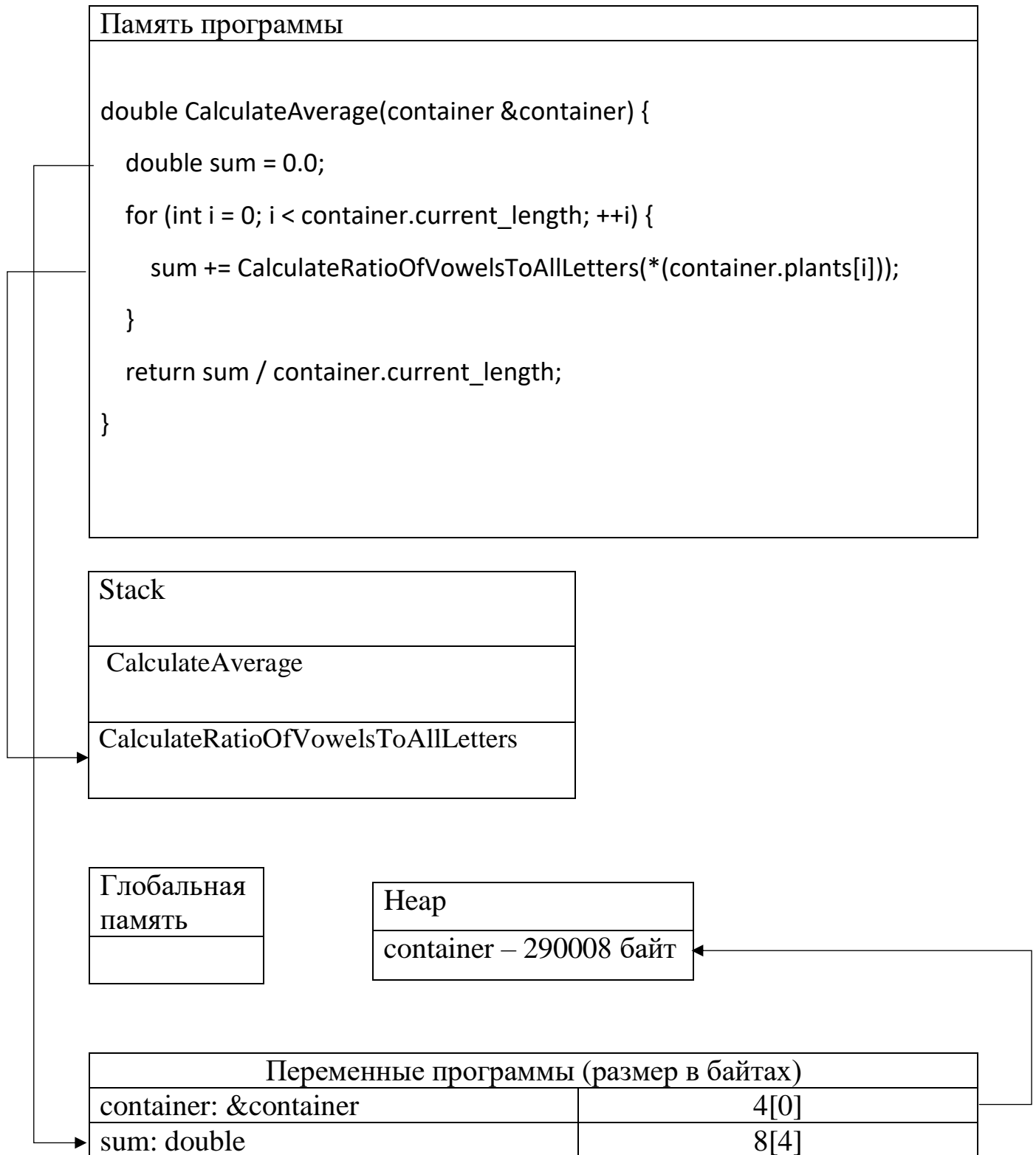
argv: int	4[0]
argc: char**	4[4]
container: &container	4[8]
size: int	4[12]
in_file: FILE*	4[16]
out_file1: FILE*	4[20]
out_file2: FILE*	4[24]

Глобальная
память

Heap

container – 290008
байт

2. Схема памяти функции по подсчёту среднего значения.



3. Схема памяти функции по модификации контейнера в соответствии с вариантом 19.

Память программы

```
void ModifyContainer(container &container) {  
    double average = CalculateAverage(container);  
    int i = 0;  
  
    while (i < container.current_length) {  
        if (CalculateRatioOfVowelsToAllLetters(*container.plants[i]) < average) {  
            for (int j = i; j < container.current_length - 1; j++) {  
                container.plants[j] = container.plants[j + 1];  
            }  
            --container.current_length;  
        } else {  
            ++i;  
        }  
    }  
}
```

Stack

ModifyContainer

CalculateAverage

CalculateRatioOfVowelsToAllLetters

Глобальная память

Heap
container – 290008 байт

Переменные программы (размер в байтах)	
container: &container	4[0]
average: double	8[4]
i: int	4[12]

Таблица типов:

Тип	Размер (байт)
int	4
double	8
char[21]	21
<i>struct container</i>	290008
current_length: int	4[0]
plants: *plant[max_length]	290000[8]
int: max_length	4[4]
<i>struct plant</i>	29
enum key{}	4
plant_type: key	4[0]
name: char[21]	21[4]
union { tree tree; bush bush; flower flower; }	4[25] 4[25] 4[25]
<i>struct tree</i>	4
age: long	4[0]
<i>struct flower</i>	4
placement_type: flower_key	4[0]
enum flower_key { }	4
<i>struct bush</i>	4
flowering_month: bush_key	4[0]
enum bush_key	4
in_file: FILE*	4
out_file1: _ FILE*	4
out_file2: _ FILE*	4

argc: char**	4
argv: int	4

3. Основные характеристики программы:

Число заголовочных файлов (внутренних): 5

Число заголовочных файлов (библиотек): 4

Число модулей: 6

Общий размер исходных текстов (программы): 15.84 Кбайт

Полученный размер исполняемого кода: 23 Кбайт

Время выполнения программы для тестовых наборов данных:

№ теста	Время (в секундах)
1	0.001703
2	0.00148
3	0.002179
4	0.01543
5	0.16013
6	0.081633
7	0.014050
8	0.001601
9	0.332784