

# Отчет

Пасилецкий Даниил Олегович

ФКН ПИ БПИ202. Вариант 55. Вариант задания 13. Вариант функции 4.

Программа выполнена в процедурном стиле. На языке программирования C++ 17 стандарт. Программа содержит следующие структуры:

1. *tree* - дерево содержащие имя и возраст.
2. *bush* - кустарник содержит имя и месяц цветения.
3. *flowers* - цветок содержит имя и вид.
4. *plants* - базовый класс, в котором используется ключ для определения типа растения. 1 - *tree*; 2 - *bush*; 3 - *flowers*
5. *container* - место для хранения массива растений.

Согласно варианту 13 задания. Так же реализована шейкерная сортировка (Shaker Sort).

## Запуск программы

Запуск программы производится через консоль с указанием специальных аргументов:

*-f infile outfile01 outfile02* - ввод из файла *infile*, а вывод в *outfile01* - содержимое контейнера, а в *outfile02* - контейнер после сортировки.

*-n number outfile01 outfile02* - создание растений рандомом, *number* - число созданных растений. Вывод в *outfile01* - содержимое контейнера, а в *outfile02* - контейнер после сортировки.

Ввод в программу через файл осуществляется следующим способом:

- В начале вводится код элемента 1 - tree; 2 - bush; 3 - flowers.
- Затем через пробел (перенос строки) имя.
- Также через пробел индивидуальная характеристика в следующем формате:
  - для дерева количество лет (целое число)
  - для кустарника месяц цветения (целое число от 1 до 12 включительно)
  - для цветка его вид, так же целое число где:
    - 1 - домашний
    - 2 - садовый
    - 3 - дикий

После всего пустая строка НЕ должна быть




## Вывод программы

Программа выводит файлы в 2 файла указанные при запуске, в первом файле находится не отсортированный массив, где указаны характеристики каждого растений. Во втором файле находится уже отсортированный массив.




С целью демонстрации двух способов вывода перечисления, в flowers оно выводится словами, а в bush номером месяца.

## Таблица типов




container

 Имя	 тип	 размер
<u>len</u>	int	4
<u>cont</u>	plants *	120*10 001 = 1 200 120




## plants

 Name	 Тип	 Размер
<u>k</u>	key (enum)	4
<u>t</u>	tree	40
<u>b</u>	bush	38
<u>f</u>	flowers	38




## tree

 Name	 Тип	 Размер
<u>age</u>	long int	8!
<u>name</u>	string	32!!

## bush

 Name	 Тип	 Размер
<u>monthes</u>	Monthes (enum)	4
<u>name</u>	string	32!!




## flowers

 Name	 Тип	 Размер
<u>type</u>	Type(enum)	4
<u>name</u>	string	32!!

8! - при x64




32!! - Различные реализации могут выделять разные объемы памяти при построении по умолчанию но как правило gcc / Linux / ARM64 / libstdc ++: 32 байта. Так же это зависит от количество символов в строке.

## Глобальная память




 Name	 Тип	 размер
<u>max_len</u>	int max_len	4

## Память программы




**int main(int argc, char\* argv[])**




 Name	 тип	 Размер
<u>argc</u>	int	4[0]
<u>argv</u>	char*	8[4]
<u>c</u>	container	1 200 124[12]
<u>size</u>	int	4[1 200 136]

**void QuotientSort(container &c)**




 Name	 тип	 Размер
<u>control</u>	int	4[0]
<u>left</u>	int	4[4]
<u>right</u>	int	4[8]
<u>i</u>	int	4[12]
<u>i</u>	int	4[16]

**plants \*InRnd()**

 Name	 тип	 Размер
<u>plant</u>	plants	120[0]

 Name	 тип	 Размер
<u>k</u>	int	4[120]

#### Временные показатели на тестах (использовался рандом)

 Name	 number	 time
<u>Test_1</u>	10	0.028s
<u>Test_2</u>	100	0.031s
<u>Test_3</u>	1000	0.039

Использовалась системное время

