

Домашнее задание №2

Колтунов Кирилл, БПИ207

1. Описание задания:

Разработать программный продукт с использованием процедурного подхода и статической типизацией. Программа должна содержать следующие структуры и функции:

Обобщенный артефакт, используемый в задании	Базовые альтернативы (уникальные параметры, задающие отличительные признаки альтернатив)	Общие для всех альтернатив переменные	Общие для всех альтернатив функции
Растения	1. Деревья (возраст – длинное целое) 2. Кустарники (месяц цветения – перечислимый тип) 3. Цветы (домашние, садовые, дикие... – перечислимый тип)	Название – строка символов.	Частное от деления числа гласных букв в названии на общую длину названия (действительное число)

Дополнительная функция:

Упорядочить элементы контейнера по убыванию используя сортировку с помощью прямого слияния (Straight Merge). В качестве ключей для сортировки и других действий используются результаты функции, общей для всех альтернатив (процент гласных букв).

2. Структурная схема архитектуры ВС с программой:

Название	Размер
int	4
double	8
FILE	216
char*	8
enum key {}	4
class Container: length : int Plant: *containers [10000]	48004 4[0] 48000[4]
class Plant: rnd20 : Random Ссылки на виртуальные функции.	8[0] 8[4]
class Flower: name : char* type : enum type { Home : type Garden : type	20 4[20] 4[24] 4[28]

Wild : type }	4[32]
class Shrub: name : char* month : enum month { January February March April May June July August September October November December };	20 4[20] 4[24] 4[28] 4[32] 4[36] 4[40] 4[44] 4[48] 4[52] 4[56] 4[60] 4[64] 4[68]
class Tree: name : char* age : unsigned long int	20 32[20]
class Random: first : int last : int	4[0] 4[4]

3. Описание работы функции main в рамках архитектуры:

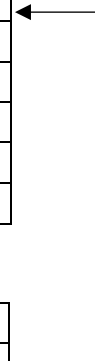
Stack
In (InRnd)
Out
StraightMerge
Out
Clear

Глобальная память
0

Память программы
int main(int argCount, char* args[]) {...}

Heap
"main" – 0
"-f" – 1
"input1.txt" – 2
"outfile01.txt" - 3
"outfile02.txt" - 4

Память данных	
argCount : int	4[0]
args : char*	8[4]
container : container	48004[20]
inputFile : FILE	216[48024]
outputFile1 : FILE	216[48244]
outputFile2 : FILE	216[48460]



4. Описание функции Clear в рамках архитектуры:

Stack
Clear

Глобальная память
0

Память программы
<pre>void Container::Clear() { for (int i = 0; i < length; ++i) { delete containers[i]; } length = 0; }</pre>

Heap
container[0]
.
.
.
Container[10000]

Память данных	
i : int	4[0]
length : int	4[4]
container : container	48004[20]

5. Сравнительный анализ

ООП хорошо применяется в практике программирования для более лёгкого создания управляемых проектов. В объектно-ориентированной программе модули в виде объектов взаимодействуют **путем отправки сообщений другим объектам**.

Процедурный подход подразумевает написание программного кода без использования объектов. Процедурное программирование заключается в написании кода **с или без подпрограмм**. В процедурной программе модули взаимодействуют посредством состояния чтения и записи, которое хранится в общих структурах данных.

В мире ООП объекты являются основным предметом интереса. Объект состоит из данных и кода, которому разрешено воздействовать на эти данные, и они очень тесно связаны. Это концепция инкапсуляции, скрытия информации. **Главным преимуществом, которое используется в моей программе – полиморфизм** (способность объекта использовать методы производного класса, который не существует на момент создания базового), к сожалению, аналог чего-то подобного попросту отсутствует при процедурном подходе. Это значительно упрощает код и его понимание становится более прозрачным.

6. Основные характеристики программы:

- Количество заголовочных файлов: 6
- Количество модулей реализации: 6
- Общий размер исходных текстов: 878 строк
- Размер исполняемого кода: 31,2КБ

- Время выполнения программы для различных тестовых прогонов:

Номер теста	Время выполнения, сек
1	0.0009
2	0.0006
3	0.0105
4	0.0005
5	0.0003