Ульяненков Ярослав БПИ204

**Домашнее задание №2**

Статически типизированная архитектура ВС, ориентированная на объектно-ориентированный подход

1. Описание задания:

|  |  |  |  |
| --- | --- | --- | --- |
| 11. Различные числа | 1. Комплексные (действительная и мнимая части – пара действительных чисел) 2. Простые дроби (числитель, знаменатель – пара целых чисел) 3. Полярные координаты (угол [радиан] – действительное; координаты конечной точки на плоскости) | - | Приведение каждого значения к действительному числу, эквивалентному записанному. Например, для комплексного числа осуществляется по формуле: sqrt(d^2+i^2)), а для полярных координат - расстояние. |

9. Упорядочить элементы контейнера по возрастанию используя сортировку с помощью прямого слияния (Straight Merge). В качестве ключей для

сортировки и других действий используются результаты функции, общей для

всех альтернатив.

1. Структурная схема программы:

|  |  |
| --- | --- |
| **Таблица типов** | |
| Типы | Объём памяти, байт |
| class complex:number  int x;  int y; | 8  4[0]  4[4] |
| class polar:number  int x;  int y; | 8  4[0]  4[4] |
| class fraction:number  int x;  int y; | 8  4[0]  4[4] |
| class number | 0 |
| сlass container  enum max\_len;  int len;  number \*cont[max\_len]; | 120008  4[0]  4[4]  120000[8] |

|  |  |
| --- | --- |
| **Память программы** | |
| Типы | Объём памяти, байт |
| Main(…)  Int argc;  char \*argv[];  container c;  FILE inF;  FILE f; | 120452  4[0]  8[4]  120008[12]  216[120020]  216[120236] |
| Sort(…)  int left\_border;  int right\_border;  int divider; | 12  4[0]  4[4]  4[8] |
| Merge(…)  Int left\_border;  int divider;  int right\_border;  int l;  int m;  int r;  number \*temp; | 36  4[0]  4[4]  4[8]  4[12]  4[16]  4[20]  12[24] |

Стек вызова

|  |
| --- |
| Main |
| Init |
| In– GetNumber – In |
| Out – Out – Out |
| MergeSort – Sort - Merge |
| Out – Out – Out |

1. Основные характеристики программы

Размер исходных текстов – 53кб.

Число интерфейсных модулей – 6.

Число модулей реализации – 6.

Размер исполняемого файла – 63кб.

Запуск программы производится следующей командой:

~$ <executable> <test> <output>

Где <executable> - имя исполняемого файла, <test> - путь до тестового файла, <output> - путь до файла, куда программа запишет результат своей работы.

Тестовые наборы являются текстовыми файлами, имеющими следующий формат: на первой строке размещено число n – количество элементов, после чего идёт n строк по три числа на каждой. Первое число – тип (1 – комплексное число, 2 – дробь, 3 – полярное число), второе и третье числа – числа, задающие требуемый тип.

|  |  |  |  |
| --- | --- | --- | --- |
| Номер тестового набора | Количество элементов | Используемая память, килобайт | Время работы, миллисекунды |
| 0 | 843 | 480 | 5.31 |
| 1 | 4556 | 624 | 21.7 |
| 2 | 8281 | 680 | 32.4 |
| 3 | 4219 | 568 | 19.7 |
| 4 | 8341 | 716 | 35.7 |
| 5 | 3160 | 588 | 20.2 |
| 6 | 6206 | 676 | 27.6 |
| 7 | 9518 | 752 | 37.1 |
| 8 | 8247 | 712 | 33.7 |
| 9 | 2988 | 564 | 16.18 |
| 10 | 0 | 432 | 1.2 |

Для замера времени работы использовалась библиотека chrono. Для замера памяти использовалась стандартная утилита macOS «Мониторинг системы».

1. Сравнение с предыдущей реализацией

В сравнении с процедурным подходом объектно-ориентированный проще с точки зрения организации кода и реализации задачи, однако, как показывают тесты, программа стала занимать больше памяти компьютера и медленнее работать. Впрочем, выигрыш от времени и удобства написания кода в данном случае выше, чем проигрыш от скорости работы и объёма занимаемой памяти.