Маринченко Игорь БПИ207

**Домашнее задание №4**

Архитектура ВС уровня системы (набора) команд

1. **Описание задания:**

|  |  |  |  |
| --- | --- | --- | --- |
| 7. Фильмы | 1. Игровой (режиссер - строка символов)  2. Мультфильм (способ создания – перечислимый тип = рисованный, кукольный, пластилиновый…)  3. Документальный фильм (длительность в минутах – целое) | 1. Название фильма –строка символов.  2. Год выхода -целое | Частное от деления года выхода фильма на количество символов в названии (действительное число) |

1. **Структурная схема программы:**

|  |  |
| --- | --- |
| **Таблица модулей** | |
| Модули | Объём памяти, байт |
| main.asm | 3,9 кб |
| input.asm | 3,6 кб |
| inrnd.asm | 3,9 кб |
| func.asm | 1,5 кб |
| output.asm | 4,1 кб |

1. **Основные характеристики программы**

Размер исходных текстов – 16,9кб.

Число модулей реализации – 5.

Размер исполняемого файла – 26,2кб.

**Формат входных данных:**

В начале идет число (1,2 или 3), которое определяет тип фильма, затем, на следующей строке через пробел перечисляются параметры:

1. Художественный фильм:

* Название фильма.
* Год выпуска.
* Имя режиссёра.

1. Мультипликационный фильм:

* Название фильма
* Год выпуска
* Тип анимации:

1: рисованный

2: кукольный

3: пластилиновый

1. Документальный фильм:

* Название фильма
* Год выпуска.
* Длина фильма.

Файлы test1-test3 содержат примеры вводных данных.

Файлы out[n] содержат выходные данные после формирования контейнера, а также рассчитанные значения площади для каждой из фигур.

|  |  |  |
| --- | --- | --- |
| Номер тестового набора | Количество элементов | Время работы, миллисекунды |
| 1 | 1 | 451 nsec |
| 2 | 5 | 475 nsec |
| 3 | 8 | 479 nsec |
| 4 | 1000 | 526 nsec |
| 5 | 10000 | 828 nsec |

**4. Вывод**

Архитектура уровня набора команд позволяет максимально близко взаимодействовать с процессором и системой, избавившись от всех абстракций. Как итог, программист получает наиболее эффективную реализацию программы как по времени работы, так и по количеству потребляемой памяти. Но также это увеличивает порог вхождения и время, затрачиваемое на написание программы в разы. В современном мире уже нет таких проблем с вычислительной мощностью и объемом памяти устройств, поэтому использовать архитектуру уровня набора команд нужно только там, где ей не будет альтернатив.