

Работа с файлами C++/ Многопоточность

Динамический массив

Библиотека `std::vector`

Объявление: `std::vector <int>`

Функции/методы	Описание
<code>size</code>	Текущий размер
<code>empty</code>	Наличие элементов
<code>push_back</code>	Добавить элемент в конец массива
<code>pop_back</code>	Удалить последний элемент массива
<code>swap</code>	Обменять элементы массива между собой

Доступ к файлу

Для работы с файлами требуется библиотека `<fstream>`

При работе с файлом можно выделить следующие этапы:

- 1) создать объект класса `fstream` (возможно, `ofstream` или `ifstream`);
- 2) связать объект класса `fstream` с файлом, который будет использоваться для операций ввода-вывода;
- 3) осуществить операции ввода-вывода в файл;
закрыть файл

Открытие файла

Константа	Описание
<code>ios::in</code>	открыть файл для чтения
<code>ios::out</code>	открыть файл для записи
<code>ios::ate</code>	при открытии переместить указатель в конец файла
<code>ios::app</code>	открыть файл для записи в конец файла
<code>ios::trunc</code>	удалить содержимое файла, если он существует
<code>ios::binary</code>	открытие файла в двоичном режиме

Пример использования:

```
ofstream fout("file.txt", ios::app);  
fout.open("file.txt", ios::out | ios::in);
```

```
fout.open("file.txt", ios::app);
```

Перемещение по файлу

Команда	Описание
seekg(Смещение, Позиция)	Смещение позиции ввода
seekp(Смещение, Позиция)	Смещение позиции вывода
tellg()	Текущая позиция ввода
tellp()	Текущая позиция вывода

Константа позиции	Описание
ios::beg	Начало файла
ios::curr	Текущая позиция
ios::end	Конец файла

Потоки

Для работы с файлами требуется библиотека `<thread>` или иная другая

При работе с потоком можно выделить следующие этапы:

- 1) создать функцию потока
- 2) инициализация потока
- 3) запуск потока

Работа с потоком

Объявление потока:

```
thread thr(threadFunction);
```

Функции/методы	Описание
join	Запуск потока с блокировкой вызывающего потока
detach	Запуск потока без блокировки вызывающего потока
get_id	ID текущего потока
yield	говорит планировщику выполнять другие потоки, может использоваться при активном ожидании
sleep_for	блокирует выполнение текущего потока в течение установленного периода
sleep_until	блокирует выполнение текущего потока, пока не будет достигнут указанный момент времени

Работа с блокировками

Библиотека:

`<mutex>`

Объявление потока:

`std::mutex block;`

Функции/методы	Описание
<code>lock</code>	Блокировка
<code>unlock</code>	Разблокировка