Безопасный по отношению к исключениям код

Безопасный код в смысле исключений - это код, который при выбрасывании функцией исключения работает следующим образом

- из него не течет память,
- он не завершается аварийно, если это не было предусмотрено специально,
- состояние всех объектов согласовано и известно (структуры данных не повреждены),
- еще?

Нейтральный по отношению к исключениям код - это код, передающий обработку исключения вызывающей функции.

Категории безопасности

- базовая гарантия - отсутствуют утечки памяти, даже если исключения выбрасываются.
- строгая гарантия - если операция прекращается из-за генерации исключения, состояние программы остаётся неизменным
- гарантия отсутствия исключений - исключения не выбрасываются.

Напоминание-ликбез

- исключение - это такой объект любого типа,
- операторы try, catch, throw, catch(...)
- перезапуск исключения
- спецификация исключений. Нужно ли их писать?
- terminate(), unexpected()

Требования к функциям

Конструктор

- должен быть нейтральным
- может генерировать любые исключения
- почему после исключения в конструкторе мы не говорим о несогласованном состоянии объекта?

Деструктор и операторы delete

Деструктор не должен генерировать исключения. Этому есть несколько причин:

- если мы сделаем массив объектов T, где ~T() может генерировать исключения, то что случится, когда при удалении этого массива один из деструкторов выкинет исключение? Ответ такой: память потечет!

```
class Anarc
{
  ~Anarc()
  {
    throw std::exception;
  }
}

int main()
{
  try
  {
    Anarc* mas = new Anarc[3000];
    delete[] mas;
  }
  catch() {}
  ... // какой-то код
}
```

- а главное, в Стандарте сказано, что любой код, который веделяет или удаляет массив объектов, деструкторы которых генерируют исключения приводит к undefined behavior

Хорошие практики

- выполнять сначала код, который может выкинуть исключения и только потом менять состояние объекта

для этого можно использовать вспомогательные функции, которые выполняют все операции, способные вызвать исключения; перехватывают их и передают исключения вызывающей функции

- использовать идиому RAII для управления ресурсами: каждым ресурсом владеет объект. В деструкторе этого объекта происходит освобождение ресурса.
- использовать функции, не генерирующие исключений (например, swap())

Ссылки