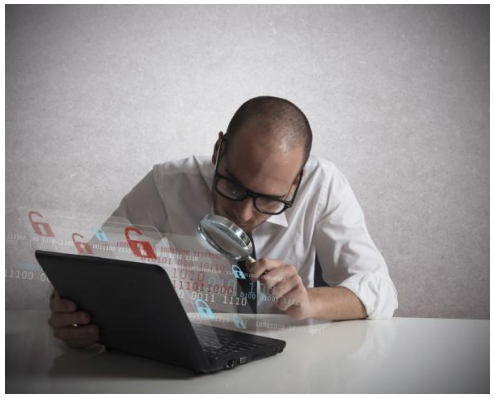


# Тема: «Программирование с «защитой от ошибок. Сквозной структурный контроль»

**Разработала**
**Топоркова Ирина Александровна**

# Вопросы лекции

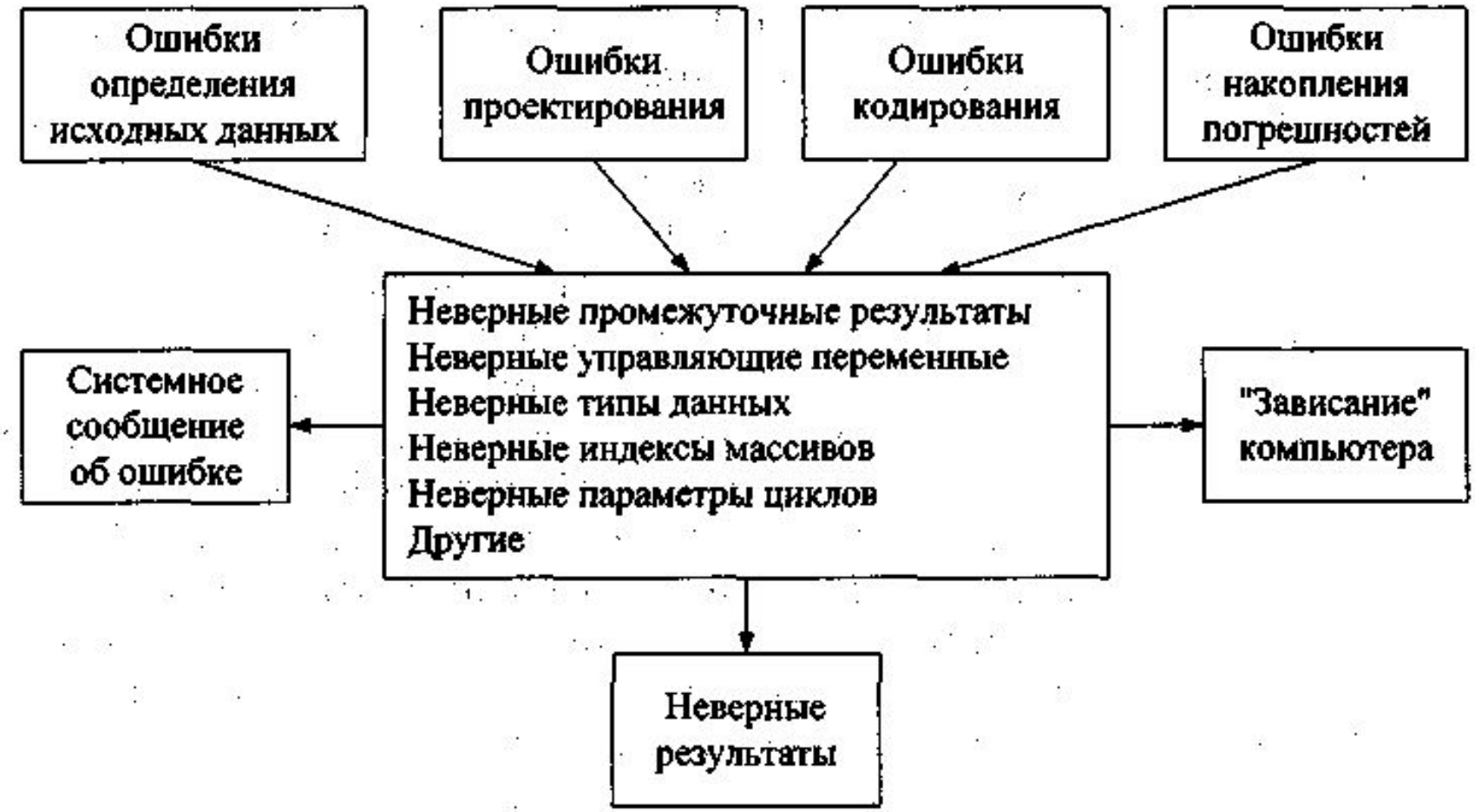
1. **Программирование с «защитой от ошибок».**
2. **Сквозной структурный контроль.**

# 1. Программирование с «защитой от ошибок»

Любая из ошибок программирования, которая не обнаруживается на этапах компиляции и компоновки программы, в конечном счете может проявиться тремя способами:

- привести к выдаче системного сообщения об ошибке;
- «зависанию» компьютера;
- получению неверных результатов.

# Способы проявления ошибок



Однако до того, как результат работы программы становится фатальным, ошибки обычно много раз проявляются в виде неверных промежуточных результатов, неверных управляющих переменных, неверных типах данных, индексах структур данных и т.п. А это значит, что часть ошибок можно попытаться обнаружить и нейтрализовать, пока они еще не привели к тяжелым последствиям.

Программирование, при котором применяют специальные приемы раннего обнаружения и нейтрализации ошибок, было названо ***защитным*** или ***программированием с защитой от ошибок*** .

При его использовании существенно уменьшается вероятность получения неверных результатов.

Детальный анализ ошибок и их возможных ранних проявлений показывает, что целесообразно проверять:

- правильность выполнения операций ввода-вывода;
- допустимость промежуточных результатов (значений управляющих переменных, значений индексов, типов данных, значений числовых аргументов и т. д.).

# Проверка правильности выполнения операций ввода-вывода

Причинами неверного определения исходных данных могут являться, как внутренние ошибки - ошибки устройств ввода-вывода или программного обеспечения, так и внешние ошибки - ошибки пользователя.

При этом принято различать:

- ***ошибки передачи*** - аппаратные средства, например, вследствие неисправности, искажают данные;
- ***ошибки преобразования*** - программа неверно преобразует исходные данные из входного формата во внутренний;
- ***ошибки перезаписи*** - пользователь ошибается при вводе данных, например, вводит лишний или другой символ;
- ***ошибки данных*** - пользователь вводит неверные данные.

# Проверка допустимости промежуточных результатов

Проверки промежуточных результатов позволяют снизить вероятность позднего проявления не только ошибок неверного определения данных, но и некоторых ошибок кодирования и проектирования. Для того чтобы такая проверка была возможной, необходимо, чтобы в программе использовались переменные, для которых существуют ограничения любого происхождения, например, связанные с сущностью моделируемых процессов.

Однако следует также иметь в виду, что любые дополнительные операции в программе требуют использования дополнительных ресурсов и могут также содержать ошибки. Поэтому имеет смысл проверять не все промежуточные результаты, а только те, проверка которых целесообразна и не сложна.

# ПРИМЕР.

- если каким-либо образом вычисляется индекс элемента массива, то следует проверить, что этот индекс является допустимым;
- если строится цикл, количество повторений которого определяется значением переменной, то целесообразно убедиться, что значение этой переменной не отрицательно;
- если определяется вероятность какого-либо события, то целесообразно проверить, что полученное значение не более 1, а сумма вероятностей всех возможных независимых событий равна 1 и т. д.

# Предотвращение накопления погрешности

Чтобы снизить погрешности результатов вычислений, необходимо соблюдать следующие рекомендации:

- избегать вычитания близких чисел (машинный ноль);
- избегать деления больших чисел на малые;
- сложение длинной последовательности чисел начинать с меньших по абсолютной величине;
- стремиться по возможности уменьшать количество операций;
- использовать методы с известными оценками погрешностей;
- не использовать условие равенства вещественных чисел;
- вычисления производить с двойной точностью, а результат выдавать с одинарной.

# Обработка исключений

Поскольку полный контроль данных на входе и в процессе вычислений, как правило, невозможен, следует предусматривать перехват обработки аварийных ситуаций.

Для перехвата и обработки аппаратно и программно фиксируемых ошибок в некоторых языках программирования, например, Delphi Pascal, C++ Java, предусмотрены **средства обработки исключений** .

Использование эти средств позволяет не допустить выдачи пользователю сообщения об аварийном завершении программы, ничего ему не говорящего. Вместо этого программист получает возможность предусмотреть действия, которые позволяют исправить эту ошибку или, если это невозможно, выдать пользователю сообщение с точным описанием ситуации и продолжить работу.

# 2. Сквозной структурный контроль

**Сквозной структурный контроль** представляет собой совокупность технологических операций контроля, позволяющих обеспечить как можно более раннее обнаружение ошибок в процессе разработки.

Термин **«сквозной»** в названии отражает выполнение контроля на всех этапах разработки.

Термин **«структурный»** означает наличие четких рекомендаций по выполнена контролирующих операций на каждом этапе.

Сквозной структурный контроль должен выполняться на специальных контрольных сессиях, в которых, помимо разработчиков, могут участвовал специально приглашенные эксперты.

Время между сессиями определяет объем материала, который выносится на сессию:

- при частых сессиях материал рассматривают небольшими порциями;
- при редких - существенным фрагментами.

Материалы для очередной сессии должны выдаваться участникам заранее, чтобы они могли их обдумать.

# Контрольные вопросы

1. От каких ошибок защищает «программирование с защитой от ошибок» и почему?
2. Что понимают под термином «исключение»?
3. В каких случаях «исключения» используют?
4. Почему «сквозной структурный контроль» так назван?
5. Что значит «сквозной» контроль?
6. В чем заключается его «структурность»?