Практическая работа №1
Тестирование и контрольный расчет программы

Цель работы:
Ознакомление с методами тестирования программного продукта;
развитие навыков составления тестов на примерах конкретных задач.

Основные теоретические сведения
При разработке программ наиболее трудоемким является этап отладки и тестирования программ. Цель тестирования, т.е. испытания программы, заключается в выявлении имеющихся в программе ошибок. Цель отладки состоит в выявлении и устранении причин ошибок.

Отладку программы начинают с составления плана тестирования. Такой план должен представлять себе любой программист. Составление плана опирается на понятие об источниках и характере ошибок. Основными источниками ошибок являются недостаточно глубокая проработка математической модели или алгоритма решения задачи; нарушение соответствия между схемой алгоритма или записью его на алгоритмическом языке и программой, записанной на языке программирования; неверное представление исходных данных на программном бланке; невнимательность при наборе программы и исходных данных на клавиатуре устройства ввода.

Нарушение соответствия между детально разработанной записью алгоритма в процессе кодирования программы относится к ошибкам, проходящим вследствие невнимательности программиста. Отключение внимания приводит и ко всем остальным ошибкам, возникающим в процессе подготовки исходных данных и ввода программы в ЭВМ. Ошибки, возникающие вследствие невнимательности, могут иметь непредсказуемые последствия, так как наряду с потерей меток и описаний массивов, дублированием меток, нарушением баланса скобок. Возможны и такие ошибки, как потеря операторов, замена букв в обозначениях переменных, отсутствие определений начальных значений переменных, нарушение адресации в массивах, сдвиг исходных данных относительно полей значений, определенных спецификациями формата.

Учитывая разнообразие источников ошибок, при составлении плана тестирования классифицируют ошибки на два типа: 1 – синтаксические; 2 – семантические (смысловые).
Синтаксические ошибки – это ошибки в записи конструкций языка программирования (чисел, переменных, функций, выражений, операторов, меток, подпрограмм).
Семантические ошибки – это ошибки, связанные с неправильным содержанием действий и использованием недопустимых значений величин.
Обнаружение большинства синтаксических ошибок автоматизировано в основных системах программирования. Поиск же семантических ошибок гораздо менее формализован; часть их проявляется при исполнении программы

в нарушениях процесса автоматических вычислений и индицируется либо выдачей диагностических сообщений рабочей программы, либо отсутствием печати результатов из-за бесконечного повторения одной и той же части программы (зацикливания), либо появлением непредусмотренной формы или содержания печати результатов.

В план тестирования обычно входят следующие этапы:

1. Сравнение программы со схемой алгоритма.
2. *Визуальный контроль программы* на экране дисплея или визуальное изучение распечатки программы и сравнение ее с оригиналом на программном бланке. Первые два этапа тестирования способны устранить больше количество ошибок, как синтаксических (что не так важно), так и семантических (что очень важно, так как позволяет исключить их трудоемкий поиск в процессе дальнейшей отладки).
3. *Трансляция программы на машинных язык*. На этом этапе выявляются синтаксические ошибки. Компиляторы с языков Си, Паскаль выдают диагностическое сообщение о синтаксических ошибках в листинге программы (листингом называется выходной документ транслятора, сопровождающий оттранслированную программу на машинном языке – объектный модуль).
4. *Редактирование внешних связей и компоновка программы*. На этапе редактирования внешних связей программных модуле программа-редактор внешних связей, или компоновщик задач, обнаруживает такие синтаксические ошибки, как несоответствие числа параметров в описании подпрограммы и обращении к ней, вызов несуществующей стандартной программы. например, 51 H вместо 51 N, различные длины общего блока памяти в вызывающем и вызываемом модуле и ряд других ошибок.
5. *Выполнение программы.* После устранения обнаруженных транслятором и редактором внешних связей (компоновщиком задач) синтаксических ошибок переходят к следующему этапу – выполнению программы на ЭВМ на машинном языке: программа загружается в оперативную память, в соответствие с программой вводятся исходные данные и начинается счет. Проявление ошибки в процессе вода исходных данных или в процессе счета приводит к прерыванию счета и выдаче диагностического сообщения рабочей программы. Проявление ошибки дает повод для выполнения отладочных действий; отсутствие же сообщений об ошибках не означает их отсутствия в программе. План тестирования включает при этом проверку правильности полученных результатов для каких-либо допустимых значений исходных данных.
6. *Тестирование программы*. Если программа выполняется успешно, желательно завершить ее испытания тестированием при задании исходных данных, принимающих предельные для программы значения. а также выходящие за допустимые пределы значения на входе.

Контрольные примеры (тесты) – это специально подобранные задачи, результаты которых заранее известны или могут быть определены без существенных затрат.

Наиболее простые способы получения тестов:

- Подбор исходных данных, для которых несложно определить результата вычислений вручную или расчетом на калькуляторе.
- Использование результатов, полученных на других ЭВМ или по другим программам.
- Использование знаний о физической природе процесса, параметры которого определяются, о требуемых и возможных свойствах рассчитываемой конструкции. Хотя точное решение задачи заранее известно, суждение о порядке величин позволяет с большой вероятностью оценить достоверность результатов.

Задания для выполнения:
Варианты заданий

1.Тестирование программного обеспечения.

Для заданного фрагмента программы (Рис.1) составить тесты, полностью охватывающие все вычислительные ветви. Произвести проверку.

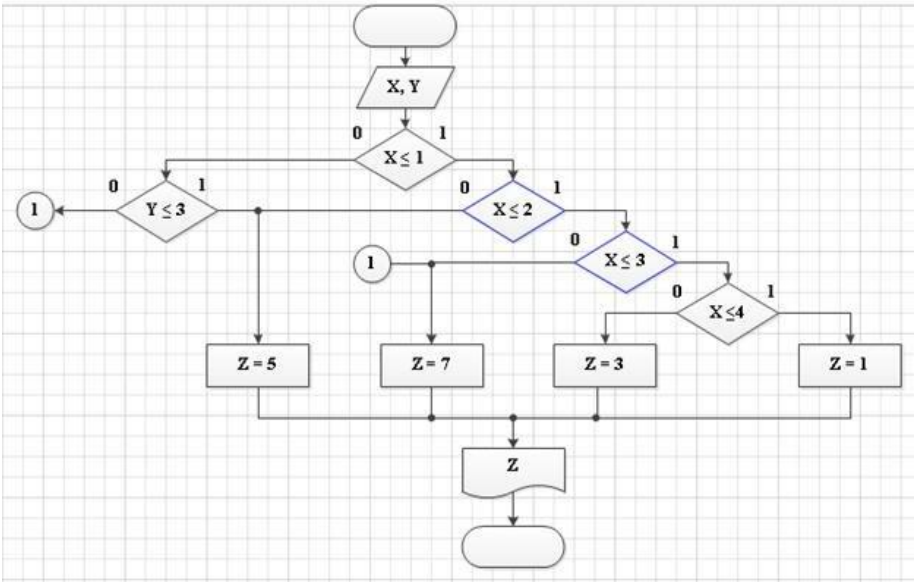

Рисунок 1 - Алгоритм программы

2.Тестирование программного обеспечения.

Для заданного фрагмента программы (Рис.2) составить тесты, полностью охватывающие все вычислительные ветви. Произвести проверку.

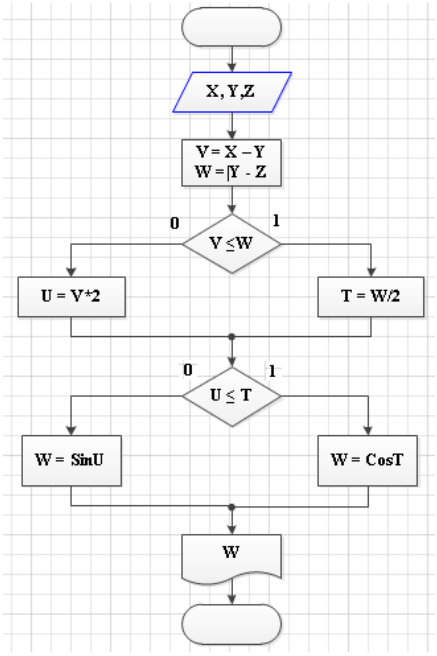

Рисунок 2 - Алгоритм программы

3.Тестирование программного обеспечения. Для заданного фрагмента программы (Рис.3) составить тесты, полностью охватывающие все вычислительные ветви. Произвести проверку.

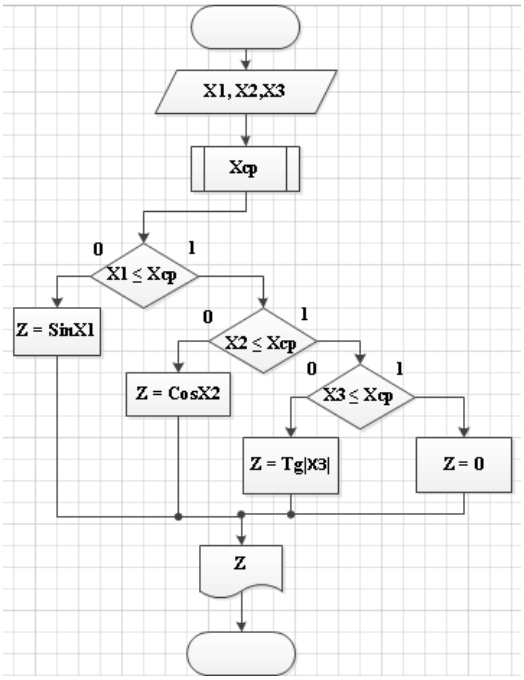

Рисунок 3 - Алгоритм программы

4.Тестирование программного обеспечения. Для заданного фрагмента программы (Рис.4) составить тесты, полностью охватывающие все вычислительные ветви. Произвести проверку.

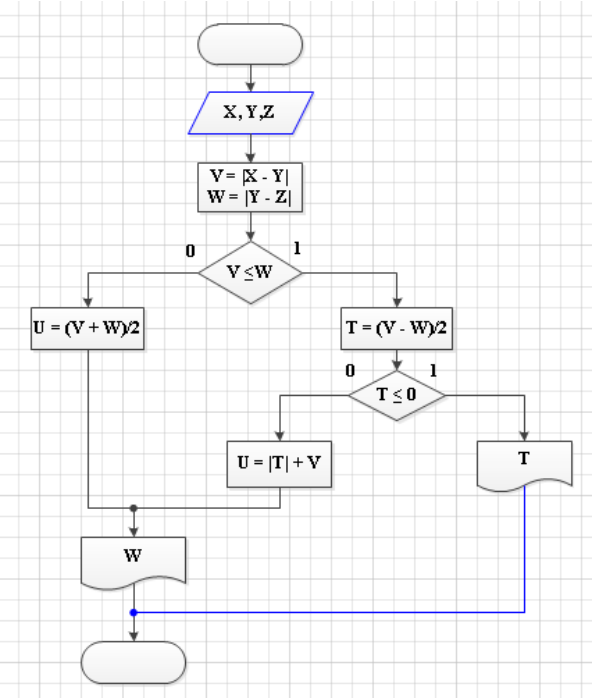

Рисунок 4 - Алгоритм программы

5.Тестирование программного обеспечения. Для заданного фрагмента программы (Рис.5) составить тесты, полностью охватывающие все вычислительные ветви. Произвести проверку.

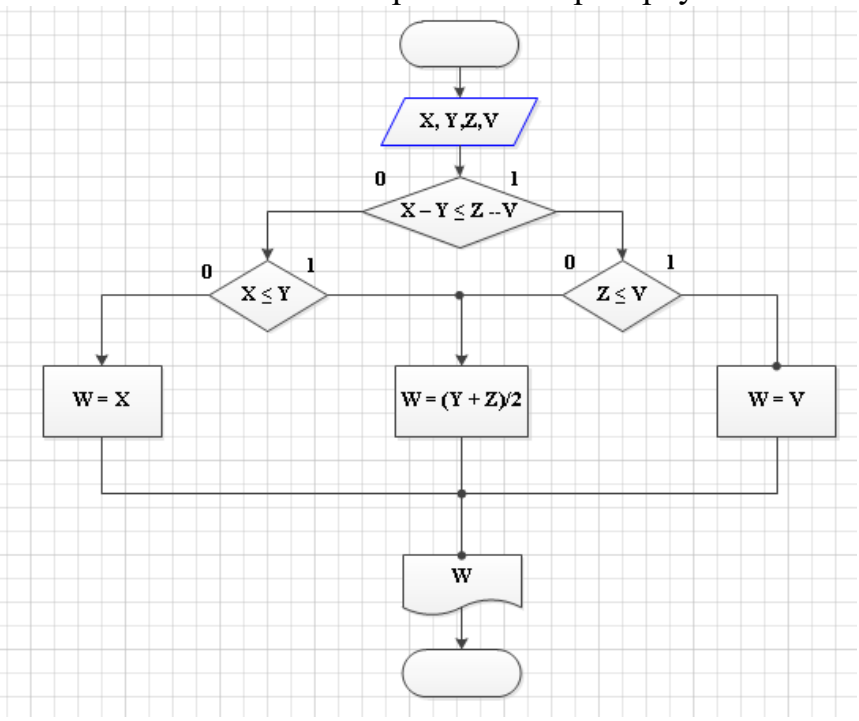

Рисунок 5 - Алгоритм программы

6.Тестирование программного обеспечения. Для заданного фрагмента программы (Рис.6) составить тесты, полностью охватывающие все вычислительные ветви. Произвести проверку.

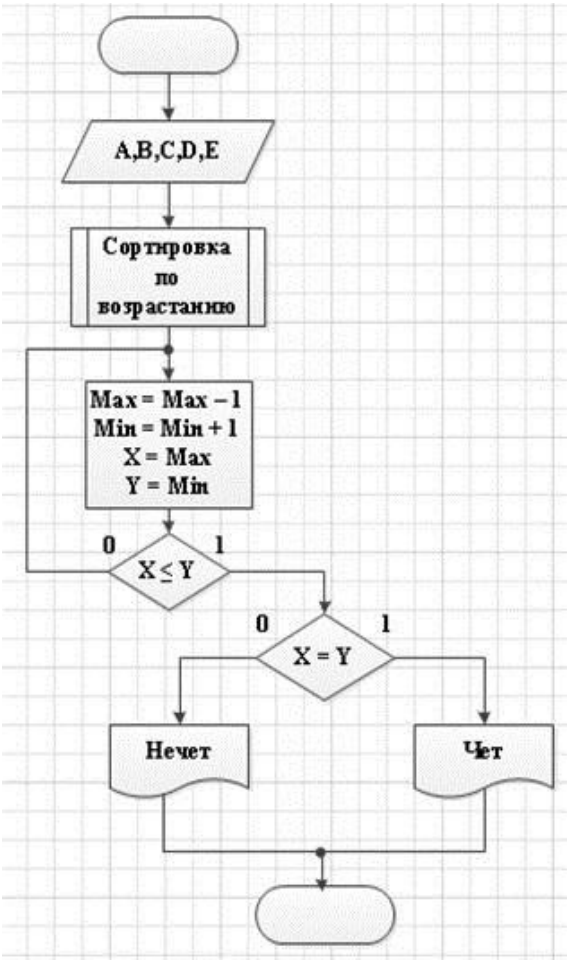

Рисунок 6 - Алгоритм программы