УВАЖАЕМЫЕ СТУДЕНТЫ! Изучите приведенную лекцию, законспектируйте основные тезисы, дайте ответы на контрольные вопросы.

Ответы на вопросы, фотоотчет, предоставить преподавателю на e-mail: r.bigangel@gmail.com до 06.02.2023.

При возникновении вопросов по приведенному материалу обращаться по следующему номеру телефона: (072)111-37-59, (Viber, WhatsApp), vk.com: https://vk.com/daykini

ВНИМАНИЕ!!! При отправке работы, не забывайте указывать ФИО студента, наименование дисциплины, дата проведения занятия (по расписанию).

Лекция № 35

Тема «Теория и методы структурного программирования»

План лекции:

1. Теория структурного программирования

2. Методы структурного программирования

1. Теория структурного программирования

К *концепциям* структурного программирования относится отказ от использования оператора безусловного перехода (***GoTo***), замена его рядом других структурированных операторов и использование идей нисходящего проектирования программ.

Основное назначение нисходящего проектирования – служить средством разбиения большой задачи на меньшие подзадачи так, чтобы каждую подзадачу можно было рассматривать независимо.

Напомним, что при использовании метода нисходящего проектирования, вначале проектируется общая структура алгоритма, без детальной проработки его отдельных частей. Затем разрабатываются блоки алгоритма, не детализированные на предыдущем шаге. В результате на каждом шаге разработки детализируется фрагмент алгоритма, то есть решается более простая задача.

В основу структурного программирования положено *требование*, чтобы каждый модуль алгоритма (программы) проектировался *с единственным входом и единственным выходом.* Программа представляется в виде

множества *вложенных* модулей, каждый из которых имеет один вход и один выход.

Базой для реализации структурированных программ является ***принцип Бома и Джакопини***, в соответствии с которым всякая реальная программа может быть построена с использованием лишь *двух управляющих конструкций*.

По Бому и Джакопини логическая структура программы может быть выражена комбинациями ***трех базовых структур***:

1)Функциональный блок;

2)Конструкция принятия двоичного (дихотомического) решения.

3)Конструкции обобщенного цикла.

Функциональный блок – это отдельный вычислительный оператор или любая другая реальная последовательность вычислений с единственным входом и единственным выходом. Изображается с помощью символа «Процесс» (рисунок 3.1).

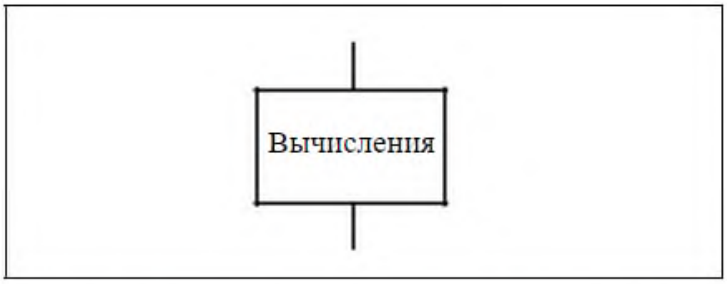

Рисунок 3.1 – Графическое представление функционального блока

Конструкция принятия двоичного (дихотомического) решения

обычно называется элементом ***If-Then-Else*** (если-то-иначе), разветвлением или ветвлением – структура, обеспечивающая выбор между двумя альтернативными путями вычислительного процесса в зависимости от выполнения некоторого условия. Изображается с помощью символов «Решение» и «Процесс» (рисунок 3.2).

Конструкция обобщенного цикла – в качестве базовой конструкции структурного программирования используется цикл с предусловием, называемый циклом «Пока» (***Do-While***). Изображается с помощью символов «Решение» и «Процесс» (рисунок 3.3).

Рисунок 3.2 – рисунок 3.3 показывают, что логические конструкции (конструкция принятия двоичного решения и конструкция обобщенного цикла) имеют только один вход и один выход. Поэтому они могут рассматриваться как функциональные блоки.

С учётом этого вводится преобразование логических блоков в функциональный блок.

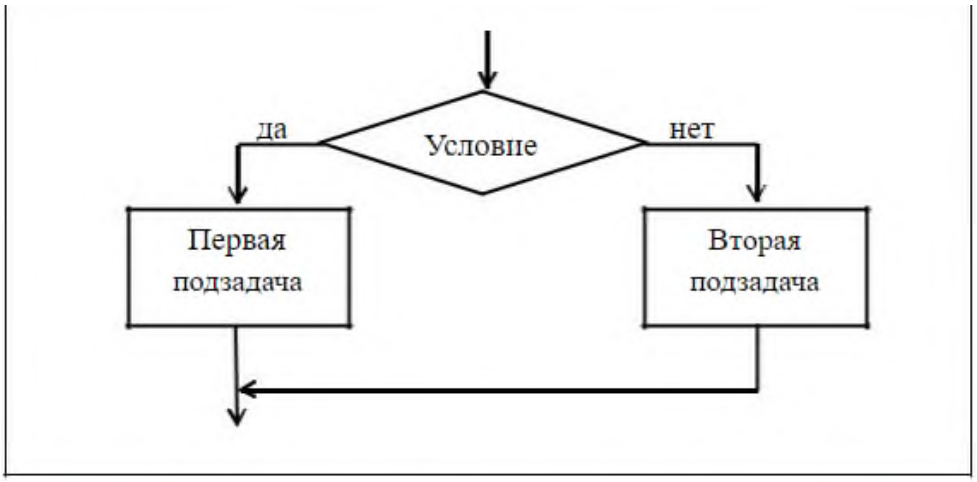

Рисунок 3.2 – Графическое представление конструкции принятия двоичного решения

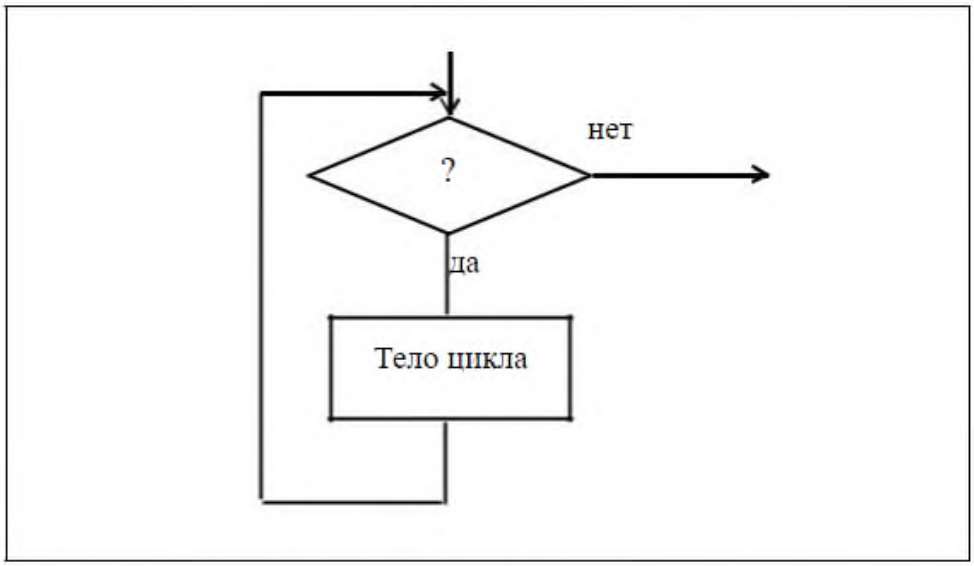

Рисунок 3.3 – Графическое представление конструкции обобщенного цикла

Кроме того, всякая последовательность функциональных элементов, называемая *конструкцией следования* (рисунок 3.4), также может быть приведена к одному функциональному блоку.

Данные преобразования называются ***преобразованиями Бома Джакопини***. Их основу составляет ***принцип "чёрного ящика"*** (что-то с одним входом и одним выходом).

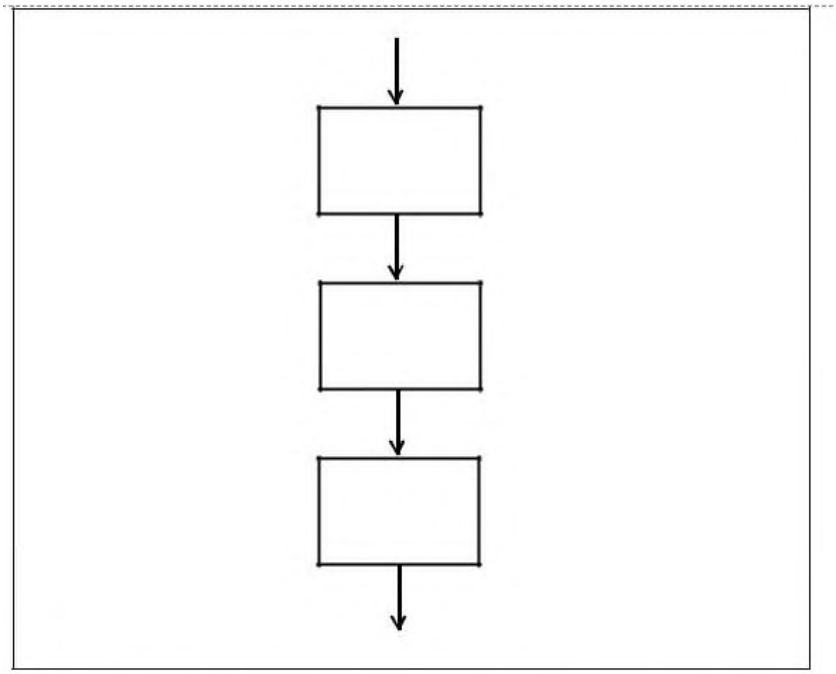

Рисунок 3.4 – Графическое представление конструкции следования

Таким образом, всякая программа (или алгоритм), состоящая из функциональных блоков, конструкций цикла и элементов If-Then-Else, поддаётся последовательному преобразованию к единственному функциональному блоку.

Эта последовательность преобразований может быть использована как средство понимания программы, *доказательство ее правильности и структурированности.*

Обратная последовательность преобразований может быть использована в процессе проектирования алгоритма (программы) по *методу нисходящего проектирования* – алгоритм (программа) разрабатывается, исходя из единственного функционального блока, который постепенно раскрывается в сложную структуру основных элементов.

2. Методы структурного программирования

Распространены две методики (стратегии) разработки программ, относящиеся к структурному программированию: программирование «сверху вниз»; программирование «снизу вверх».

Программирование «сверху вниз», или нисходящее проектирование программ, — это методика разработки программ, при которой разработка начинается с определения целей решения проблемы, после чего идет последовательная детализация, заканчивающаяся детальной программой. Сначала выделяется несколько самых глобальных задач, решение которых может быть представлено в общей структуре функционально независимыми блоками. Разработку логической структуры каждого такого блока и ее модификацию можно осуществлять независимо от остальных блоков. На этом первом этапе проекта раскрываются наиболее важные и существенные связи, определяется функциональное назначение каждого блока, его входные и выходные данные. На последующих этапах проектирования уточняется (детализируется) логическая структура отдельных функциональных блоков общей схемы, что также может осуществляться в несколько этапов детализации вплоть до простейших инструкций. На каждом этапе проекта выполняются многократные проверки и исправления.

Подобный подход является достаточно рациональным, позволяет значительно ускорить процесс разработки сложных программных проектов и в значительной мере избежать ошибочных решений. Кроме того, появляется возможность некоторые подпрограммы (модули) не реализовывать сразу, а временно отложить их разработку, пока не будут закончены другие части. Например, если имеется необходимость вычисления сложной математической функции, то выделяется отдельная подпрограмма такого вычисления, реализуется временно одним оператором, который просто присваивает нужное значение. Когда все приложение будет написано и отлажено, можно приступить к реализации этой сложной функции.

Программирование «снизу вверх», или восходящее проектирование программ, — это методика разработки программ, начинающаяся с разработки

подпрограмм (процедур, функций), в то время, когда проработка общей схемы не закончилась. Такая методика является менее предпочтительной по сравнению с нисходящим проектированием, так как часто приводит к нежелательным результатам, переписыванию кода и увеличению времени разработки. Ее использование может быть целесообразным, когда новый проект использует известные частные решения.

Общие принципы разработки программных проектов. Использование технологии структурного программирования при разработке серьезных программных проектов основано на следующих принципах:

- программирование должно осуществляться «сверху вниз»;
- весь проект должен быть разбит на модули/подпрограммы с одним входом и одним выходом;
- любая подпрограмма должна допускать только три основные структуры: последовательное выполнение операторов, ветвление и цикл;
- недопустим оператор безусловной передачи управления ***go to***;
- документация должна создаваться одновременно с программированием, частично в виде комментариев к программе.

Применение принципов и методов структурного программирования позволяет повысить надежность программ (благодаря хорошему структурированию при проектировании программа легко поддается тестированию и отладке) и их эффективность (структурирование программы позволяет легко находить и корректировать ошибки, а отдельные подпрограммы можно переделывать/модифицировать независимо от других), уменьшить время и стоимость программной разработки, улучшить читабельность программ.

Контрольные вопросы:

1. Теория структурного программирования

2. Методы структурного программирования