Дизайн и компоновка ГИС в геоэкологии

ЛЕКЦИЯ 5

ТЕМА ЛЕКЦИИ

ПРОЕКТИРОВАНИЕ ДИЗАЙНА ГИС

РАЗРАБОТКА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

В настоящее время при разработке сложного программного обеспечения обычно используют одну из двух технологий:

структурное программирование или ***объектно-ориентированное*** программирование

Первая технология для разработки сложных программ рекомендует **разбивать (декомпозировать) программу на подпрограммы (процедуры),** решающие отдельные подзадачи, т.е. базируется на **процедурной декомпозиции**

Вторая технология использует более сложный подход, при котором в предметной области задачи выделяют **отдельно функционирующие элементы**

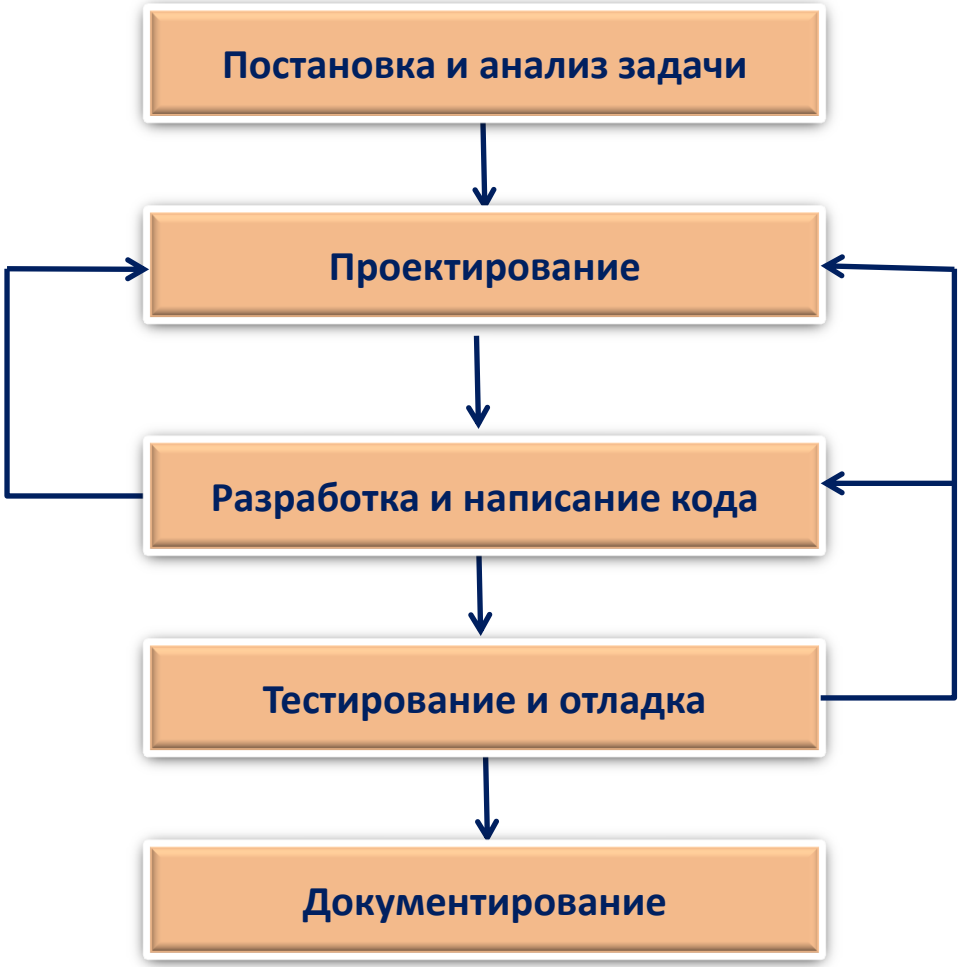
Поведение этих объектов программно моделируется с использованием специальных средств, а затем, уже из готовых объектов, опять же специальным способом, собирается сложная программа. Таким образом, в основе второй технологии лежит **объектная декомпозиция**

Современная методология разработки программного обеспечения предполагает разбиение проекта на **этапы**, на которых специалисты, играющие определенные роли в проекте, выполняют различные действия и производят составные части проекта

ЭТАПЫ РАЗРАБОТКИ

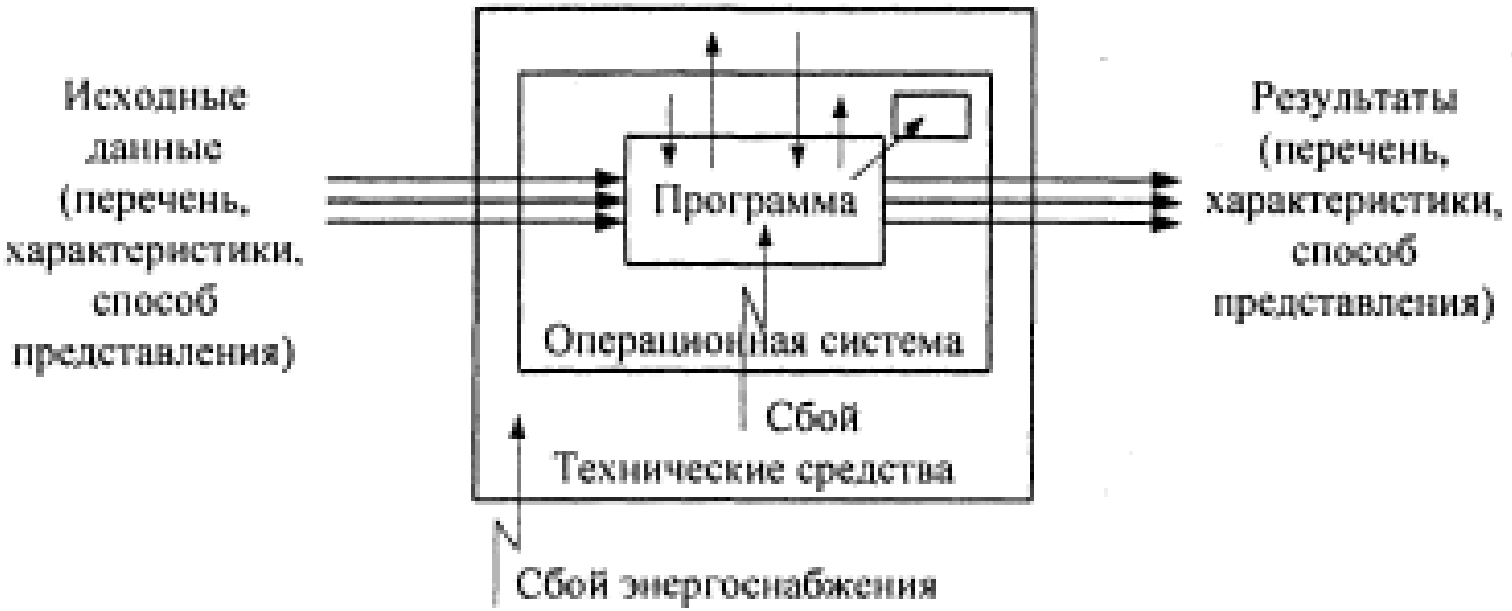
Можно выделить следующие этапы:

- Постановка и анализ задачи, определение требований
- Проектирование, разработка, написание кода;
- Тестирование, отладка и оценка качества
- Документирование
- Внедрение и сопровождение



ПОСТАНОВКА ЗАДАЧИ, ОПРЕДЕЛЕНИЕ ТРЕБОВАНИЙ

Результатом обычно является документ, называемый в нашей стране техническим заданием и содержащий сведения о назначении продукта, набор требований к нему и описание границ проекта



АНАЛИЗ ЗАДАЧИ

По результатам анализа условия задачи выбирают математические абстракции, **адекватно, т.е. с требуемой точностью и полнотой,** представляющие исходные данные и результаты, **строят модель задачи** и определяют метод преобразования исходных данных в результат **(метод решения задачи)**

ПРОЕКТИРОВАНИЕ

Принято различать логическое и физическое проектирование

Логическое проектирование не учитывает особенностей среды, в которой будет выполняться программа (технические и программные средства компьютера)

При выполнении **физического проектирования все эти параметры должны** быть учтены

ЛОГИЧЕСКОЕ ПРОЕКТИРОВАНИЕ

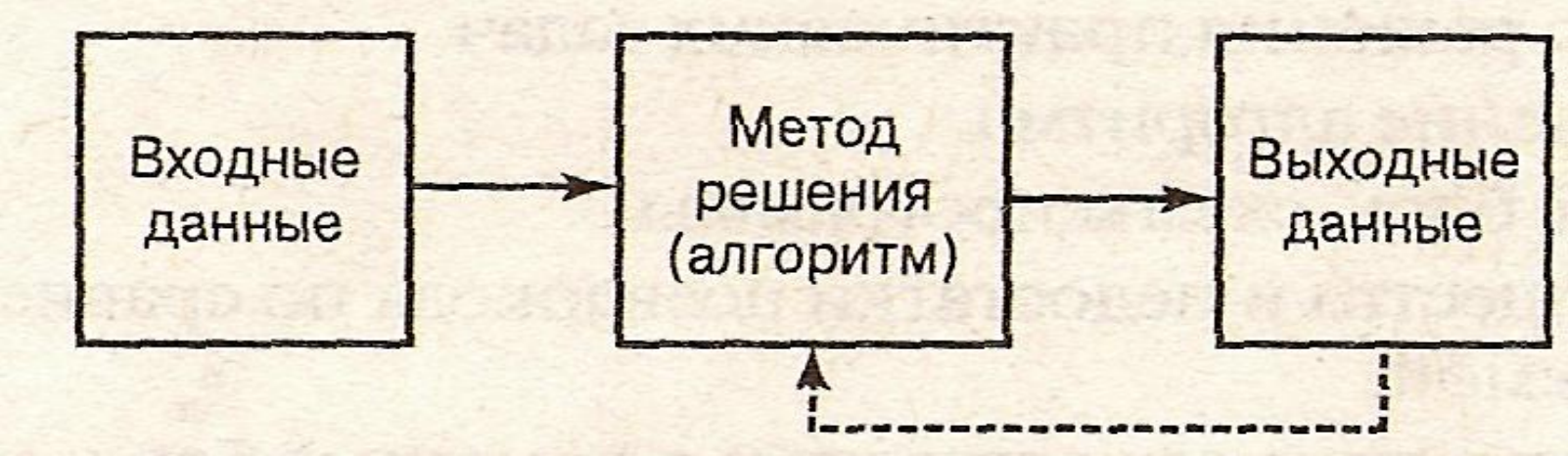
Логическое проектирование при процедурном подходе предполагает

детальную проработку последовательности действий будущей программы

Определяется

структура программы (программ) и разрабатывается алгоритм

Под **Алгоритмом** понимается точная конечная система предписаний, определяющая содержание и порядок действий **исполнителя** над некоторыми **объектами** (исходными и промежуточными данными) для получения (после конечного числа шагов) искомого результата



Любой алгоритм существует не сам по себе, а предназначен для определенного **исполнителя.** Алгоритм описывается в **командах исполнителя**, который этот алгоритм будет выполнять

Объекты, над которыми исполнитель может совершать действия, образуют так называемую **среду исполнителя**

Исходные данные и результаты любого алгоритма всегда принадлежат среде того исполнителя, для которого предназначен алгоритм

СВОЙСТВА АЛГОРИТМА

Дискретность. Выполнение алгоритма разбивается на последовательность законченных действий-шагов

Детерминированность. Способ решения задачи однозначно определен в виде последовательности шагов

Понятность. Алгоритм не должен содержать предписаний, смысл которых может восприниматься исполнителем неоднозначно

Результативность. Содержательная определенность результата каждого шага и алгоритма в целом

Свойство результативности содержит в себе свойство конечности – завершение работы алгоритма за конечное число шагов

Массовость. Алгоритм пригоден для решения любой задачи из некоторого класса задач

При изучении алгоритмов важно разделять два понятия: ***запись алгоритма* и *выполнение алгоритма***

В учебно-научной литературе термин **«алгоритм»** используется как в первом, так и во втором значении

Различают последовательности действий

- линейной структуры
- разветвленной структуры
- циклической структуры

Линейная структура процесса вычислений предполагает, что для получения результата необходимо выполнить некоторые операции в определенной последовательности

Разветвленная структура процесса вычислений предполагает, что конкретная последовательность операций зависит от значений одного или нескольких параметров

Циклическая структура процесса вычислений предполагает, что для получения результата некоторые действия необходимо выполнить несколько раз

Процессы вычислений циклической структуры в свою очередь можно разделить на **три группы**:

циклические процессы, для которых количество повторений известно ***счетные циклы или циклы с заданным количеством повторений***

циклические процессы, завершающиеся по достижении или нарушении некоторых условий - ***итерационные циклы***

циклические процессы, из которых возможны два варианта выхода: выход по завершении процесса и досрочный выход по какому-либо дополнительному условию - ***поисковые циклы***

СПОСОБЫ ЗАПИСИ АЛГОРИТМА

Способы задания:

- текстовая форма записи
- запись в виде блок-схемы
- запись алгоритма на каком-либо алгоритмическом языке
- представление алгоритма в виде машины Тьюринга или машины Поста

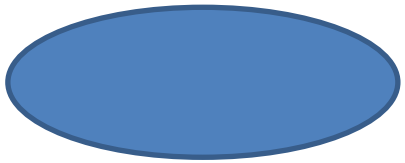
Вне зависимости от выбранной формы записи элементарные шаги алгоритма (команды) при укрупнении объединяются в алгоритмические конструкции: *последовательные, ветвящиеся, циклические, рекурсивные*

Для записи любого алгоритма достаточно трех основных алгоритмических конструкций: **последовательных, ветвящихся, циклических**

БЛОК-СХЕМА

Блок-схема алгоритма

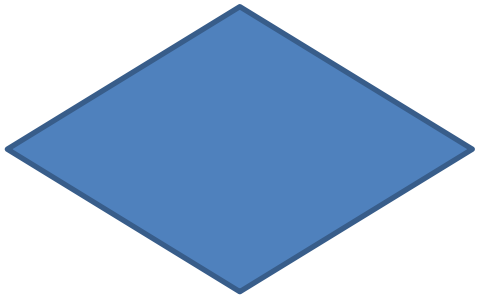
- Блок-схема алгоритма представляет собой диаграмму, на которой изображена последовательность действий, выполняемых программой



Терминатор. Показывает начальную и конечную точки программы. Терминатор соединен с другими фигурами только одной линией: из начальной точки линия со стрелкой выходит, а в конечную — входит

Ввод и вывод данных. Фрагмент программы, в котором пользователь вводит данные или программа выводит результаты на экран

Обработка данных. Отображает любую операцию, выполняемую компьютером, например присвоение переменной какого-либо значения



Структура принятия решения. Фрагмент программы, в котором принимается решение о направлении вычислительного процесса. В ромб всегда входит одна линия, а выходят две

Одна из выходящих линий отмечается словом **"Да"**, а другая — **"Нет"**

Счетные циклы

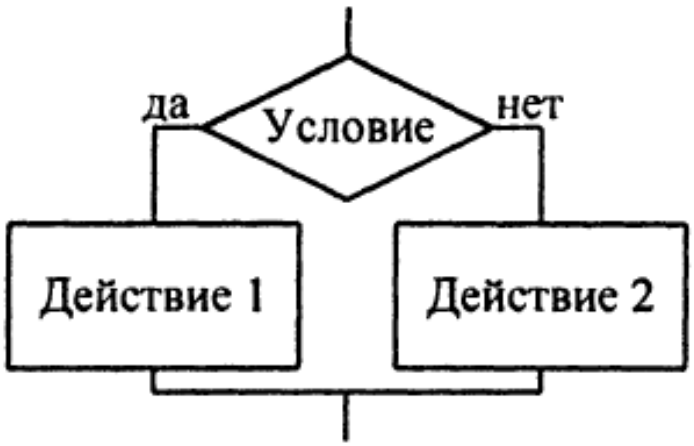
Предопределенный процесс. Эта фигура отображает группу операций, например вычисление факториала

Соединитель. С помощью этой фигуры можно избежать перекрещивания линий и сделать блок-схему менее запутанной. Соединители всегда используются парами: в один соединитель линия входит, а из другого выходит, причем в каждый из соединителей данной пары заносится одно и то же число

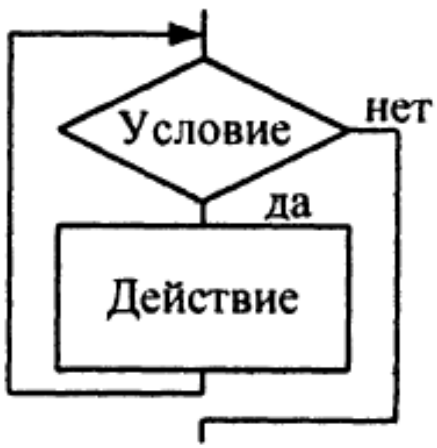
Линия. Соединяет две фигуры блок-схемы и показывает последовательность выполняемых программой операций

- **Базовые конструкции:**

а

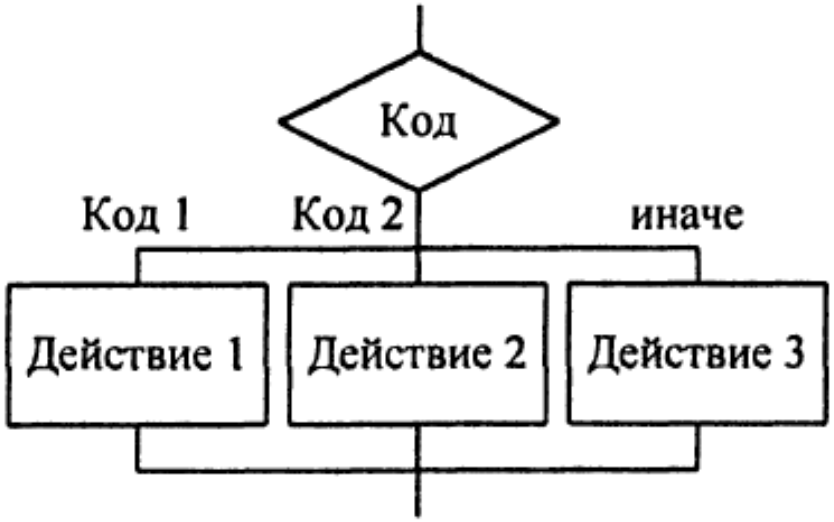


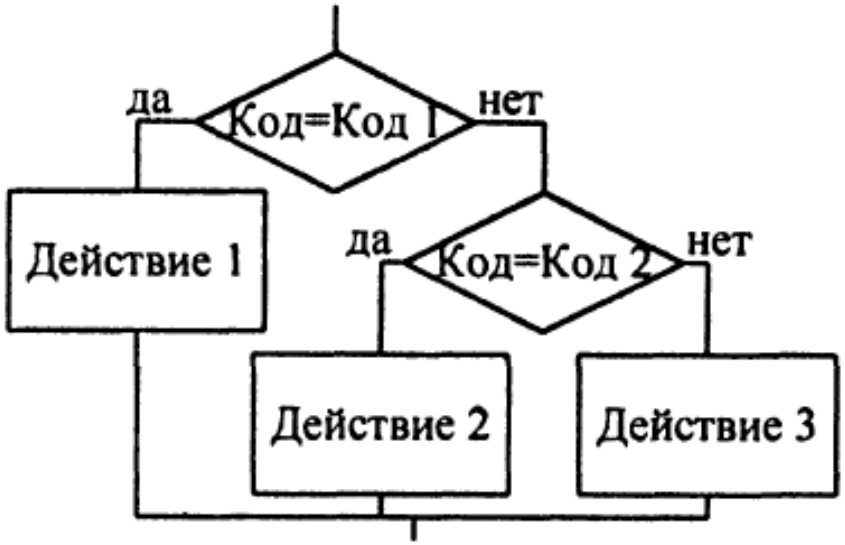
б



в

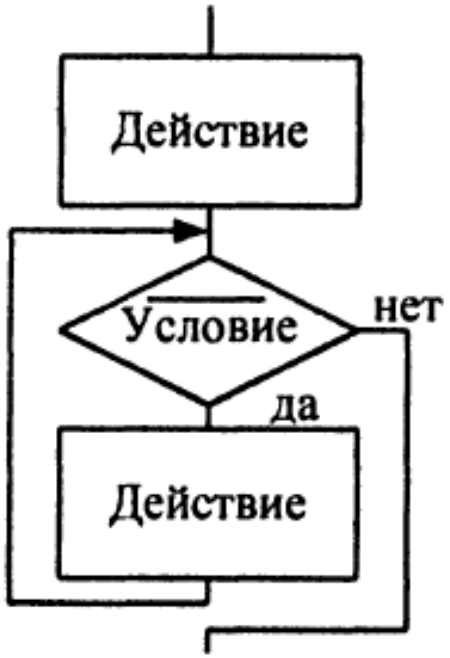
Выбор:


Код
Код 1
Код 2
иначе
Действие 1
Действие 2
Действие 3


да
Код=Код 1
нет
Действие 1
да
Код=Код 2
нет
Действие 2
Действие 3

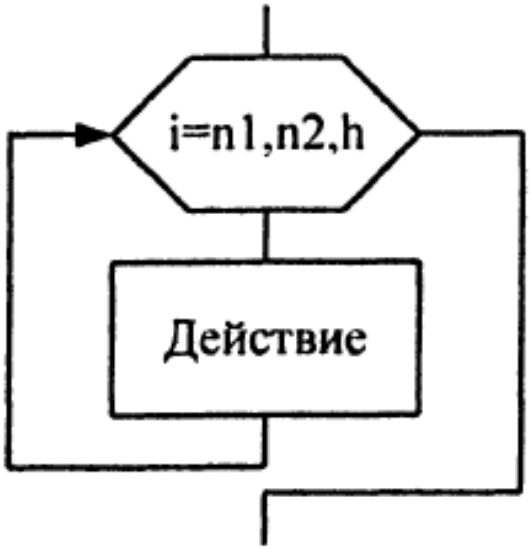
Цикл с пост условием (цикл-до)

в

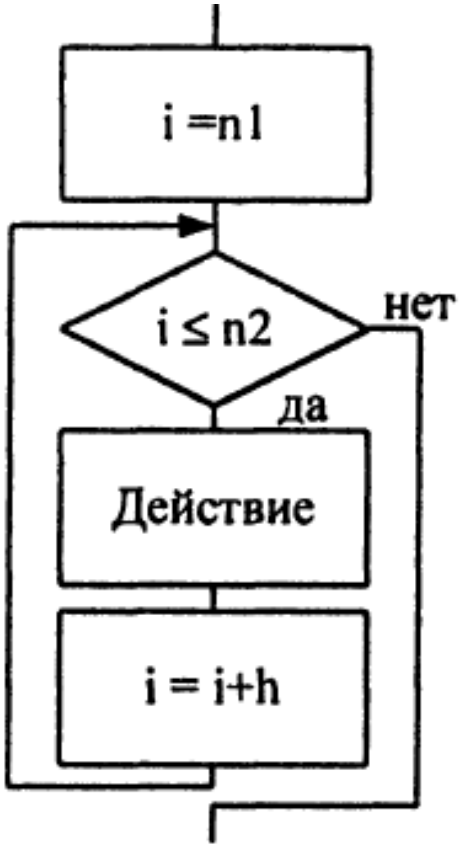


г

Цикл с заданным числом повторений



д



е

Перечисленные структуры были положены в основу **структурного программирования**

ПСЕВДОКОД

Псевдокод

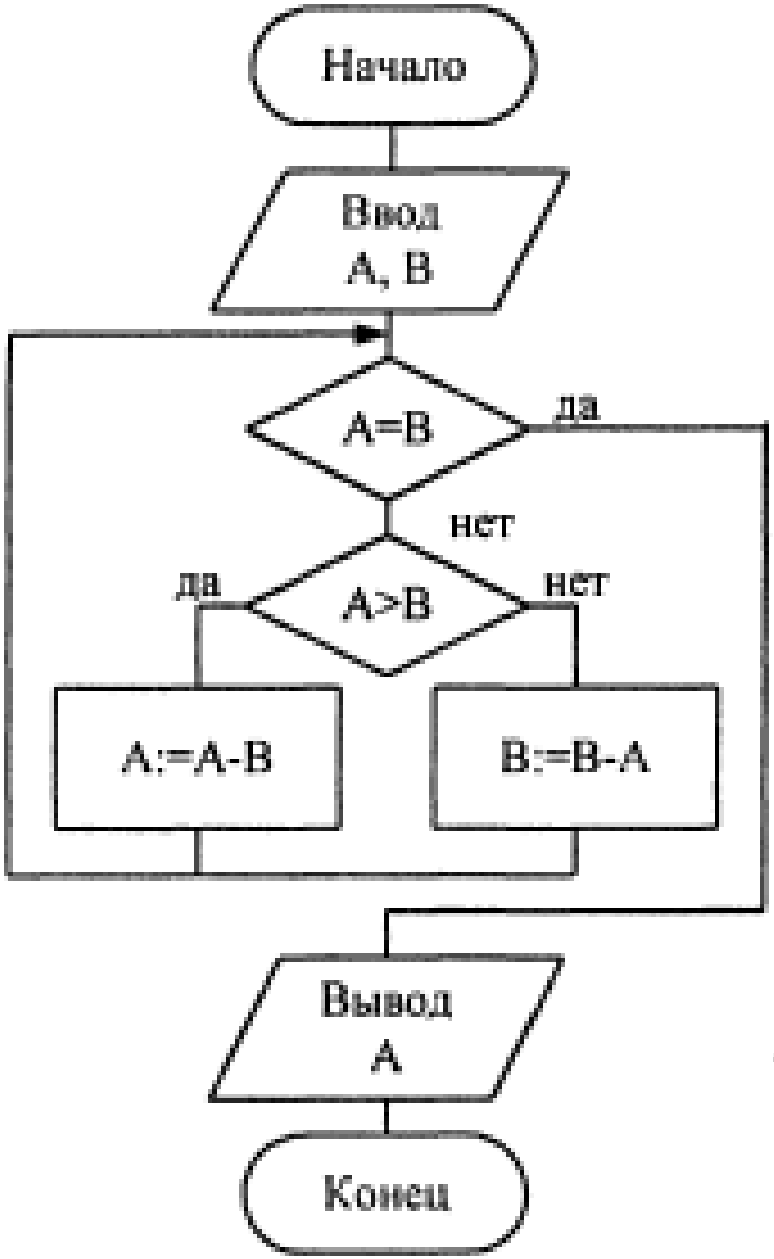
Состоит из комбинации операторов языка высокого уровня и фраз на английском или русском языке. Стандартов на составление псевдокодов не существует

БАЗОВЫЕ КОНСТРУКЦИИ

Структура	Псевдокод	Структура	Псевдокод
Следование	<действие 1> <действие 2>	Выбор	**Выбор** <код> <код1>: <действие 1> <код2>: <действие 2> ... **Все-выбор**
Ветвление	**Если** <условие> **то** <действие 1> **иначе** <действие 2> **Все-если**	Цикл с заданным количеством повторений	**Для** <индекс> = <n>,<k>,<h> <действие> **Все-цикл**
Цикл-пока	**Цикл-пока** <условие> <действие> **Все-цикл**	Цикл-до	**Выполнять** <действие> **До** <условие>

- Алгоритм Евклида:

Алгоритм Евклида:
Ввести A,B
цикл-пока A ≠ B
если A > B
то A := A - B
иначе B := B - A
все-если
все-цикл
Вывести A
Конец алгоритма.



Алгоритмы простых программ разрабатывают, продумывая последовательность действий для решения некоторой задачи

Для разработки алгоритмов более сложных программ целесообразно использовать ***метод пошаговой детализации***

С использованием данного метода разработку алгоритмов выполняют поэтапно

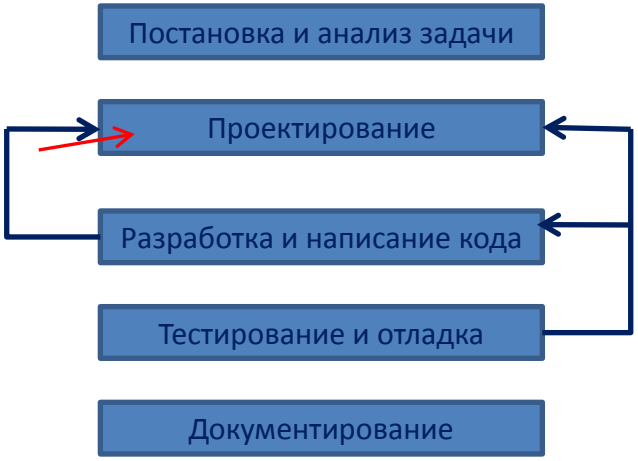
На первом этапе описывают решение поставленной задачи, выделяя подзадачи и считая их решенными

На следующем - аналогично описывают решение подзадач, формулируя уже подзадачи следующего уровня. Процесс продолжают до тех пор, пока не дойдут до подзадач, алгоритмы решения которых очевидны

При этом, описывая решение каждой задачи, желательно использовать не более одной-двух конструкций, таких как цикл или ветвление, чтобы четче представлять структуру программы

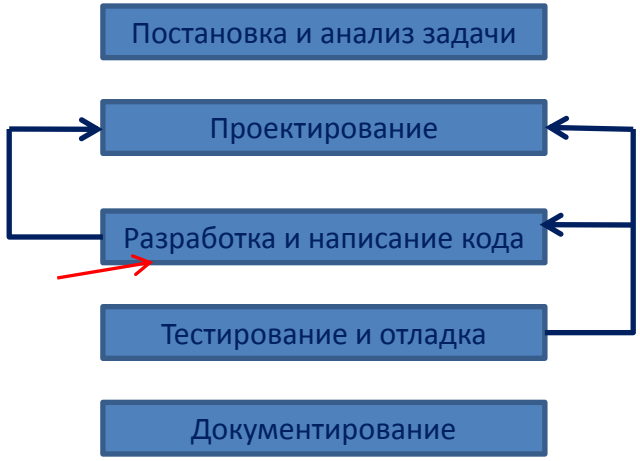
ФИЗИЧЕСКОЕ ПРОЕКТИРОВАНИЕ

Физическое проектирование



При выполнении физического проектирования осуществляют ***привязку разрабатываемого программного обеспечения*** к имеющемуся набору технических и программных средств

СОЗДАНИЕ ПРОГРАММНОГО КОДА



- Разработанные алгоритмы реализуют, составляя по ним текст программы с использованием конкретного языка программирования

- На этапе собственно разработки создается код приложения в соответствии с техническим проектом

Вначале осуществляют ввод программы в компьютер. Для ввода используют специальную программу - ***текстовый редактор***, с помощью которого создают файл, содержащий текст программы.

Программа преобразующая программу, записанную на одном из языков высокого уровня, в программу на машинном языке - **транслятор**

Трансляторы реализуются в виде

компиляторов или **интерпретаторов**

- **Интерпретатор** (*англ. interpreter — истолкователь, устный переводчик*) переводит и выполняет программу строка за строкой

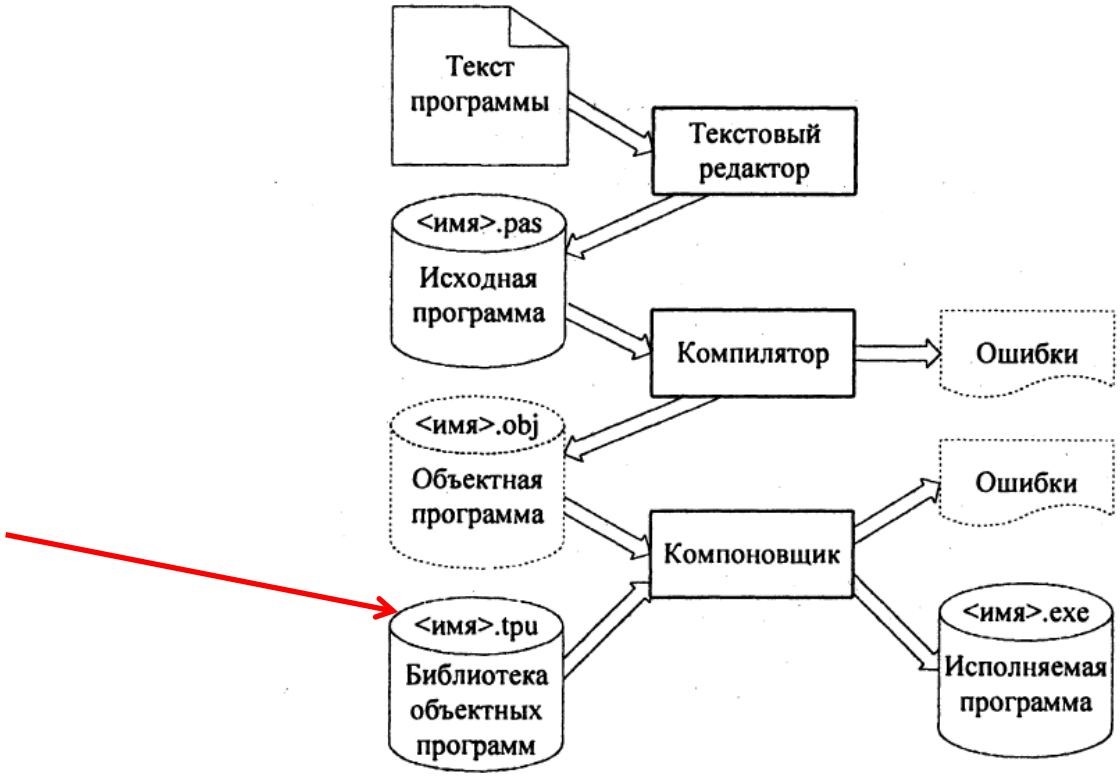
- **Компилятор** (англ. compiler — составитель, собиратель) читает всю программу целиком, делает ее перевод и создает законченный вариант программы на машинном языке, который затем и выполняется

Запускаем компилятор

В процессе разбора и преобразования программы компилятор может обнаружить ошибки. Тогда он аварийно завершает работу, выдав программисту сообщения об ошибках компиляции

После исправления ошибок процесс компиляции повторяют

Если с точки зрения компилятора программа написана правильно, то он строит так называемый ***объектный код***



- После работы компоновщика получаем исполняемую программу

На этом этапе основным инструментом, обязательным к применению, является средство разработки приложений, система программирования:

Borland Jbuilder, Borland Delphi, Borland Developer Studio, Borland C++ Builder, Microsoft Visual Basic, Microsoft Visual C++, C# , Microsoft Visual Studio, Microsoft Visual Studio.NET

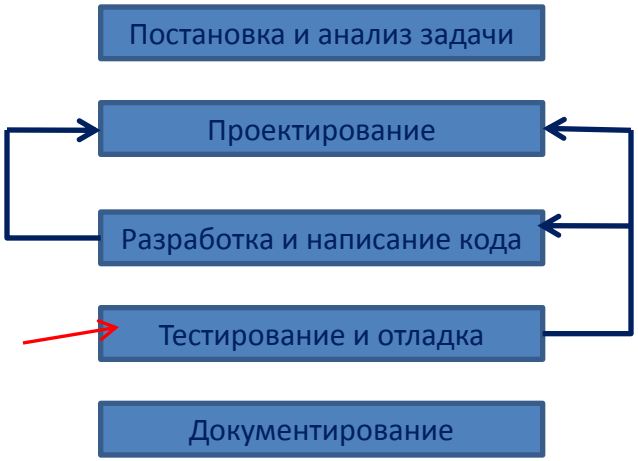
Современные системы программирования обычно предоставляют пользователям мощные и удобные средства разработки программ, интегрированные среды разработки

В них входят:

- компилятор или интерпретатор
- средства создания и редактирования текстов программ
- библиотеки стандартных программ и функци;
- отладочные программы, т.е. программы, помогающие находить и устранять ошибки в программе
- графические библиотеки; утилиты для работы с библиотекам;
- встроенный ассемблер
- встроенная справочная служба и т.д.

ТЕСТИРОВАНИЕ И ОЦЕНКА КАЧЕСТВА

- При тестировании продукта проверяется его соответствие требованиям, и в зависимости от этих требований осуществляется определение методик тестирования, создание тестов и выбор соответствующих инструментов.

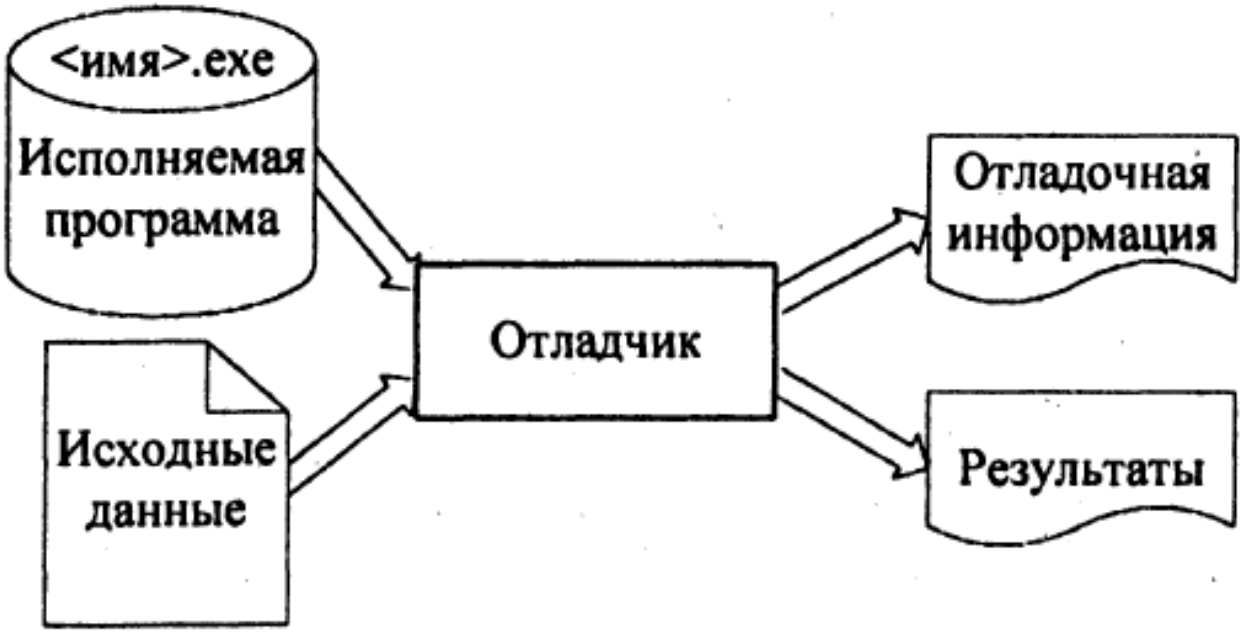


Помимо ошибок, обнаруживаемых автоматически компилятором, компоновщиком или операционной системой, существует еще группа очень опасных **логических ошибок**

Наличие таких ошибок в программе приводит к выдаче неправильных результатов. Для их обнаружения параллельно с отладкой программы осуществляют ее тестировании

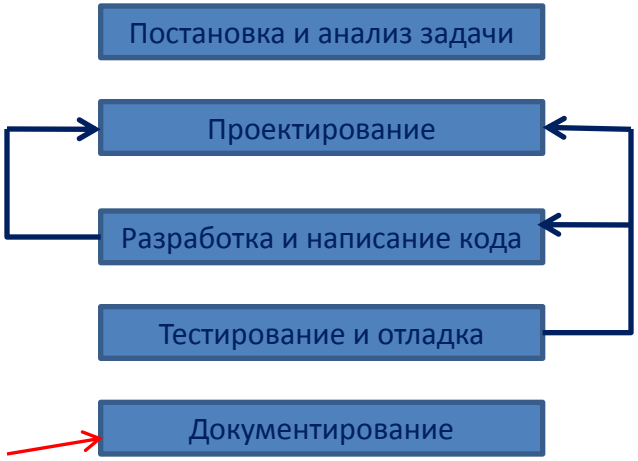
Тестированием называют процесс выполнения программы при различных тестовых наборах данных с целью обнаружения ошибок

Процесс локализации и исправления ошибок получил название **отладки** программы. При отладке программы часто используют специальные *программы-отладчики, которые позволяют выполнить любой фрагмент программы*



ДОКУМЕНТИРОВАНИЕ

- В простейшем случае файл справочной системы можно создать с помощью Microsoft Word и утилит от Microsoft для создания help-файлов, включаемых в состав многих средств разработки, но при большом объеме работы нередко используются специализированные средства таких компаний, как Blue Sky Software, EC Software, JGsoft



- Главное назначение документации — позволить человеку, не являющемуся разработчиком программы, использовать ее и при необходимости модифицировать ее код

ВНЕДРЕНИЕ И СОПРОВОЖДЕНИЕ

- Для создания дистрибутивных приложения также применяются специализированные средства, лидерами рынка которых являются компании InstallShield Software и Wise Solutions

- На этапе сопровождения продукта, как показывает практика, может понадобиться все, что было произведено при работе над проектом, и соответственно любой из инструментов