АЛГОРИТМИЗАЦИЯ И ПРОГРАММИРОВАНИЕ

и.о. доцент кафедры «Информационные системы»
Муханова Аягоз Асанбековна

Лекция №1 **Введение в программирование на языке Python.**
Понятия «алгоритм» и «программа».

План лекции:

1. Понятия «алгоритм» и «программа»
2. Знакомство с языком программирования Python
3. История языка Python

Основные понятия и определения (глоссарий)

Алгоритм — точное предписание исполнителю совершить определённую последовательность действий для достижения поставленной цели за конечное число шагов.

Данные — сведения:

- полученные путём измерения, наблюдения, логических или арифметических операций; и представленные в форме, пригодной для постоянного хранения, передачи и (автоматизированной) обработки.

Тип данных — характеристика набора данных, которая определяет:

- диапазон возможных значений данных из набора;
- допустимые операции, которые можно выполнять над этими значениями;
- способ хранения этих значений в памяти.

Различают:

- простые типы данных: целые, действительные числа, символы, строки, логические величины;
- составные типы данных: массивы, файлы и др.

Основные понятия и определения (глоссарий)

Программа — согласно ГОСТ 19781-90 — данные, предназначенные для управления конкретными компонентами системы обработки информации в целях реализации определённого алгоритма.

Язык программирования — искусственный (формальный) язык, предназначенный для записи алгоритмов. Язык программирования задаётся своим описанием и реализуется в виде специальной программы: компилятора или интерпретатора.

Основные понятия и определения (глоссарий)

- **Транслятор**

Транслятор — в широком смысле — программа, преобразующая текст, написанный на одном языке, в текст на другом языке.

Транслятор — в узком смысле — программа, преобразующая: программу, написанную на одном (входном) языке в программу, представленную на другом(выходном) языке.

Транслятор языка программирования — программа, преобразующая исходный текст программы на языке программирования в машинный язык вычислительной системы, на которой эта программ должна выполняться.

- **Интерпретатор**

Интерпретатор — транслятор, способный параллельно переводить и выполнять программу, написанную на алгоритмическом языке высокого уровня.

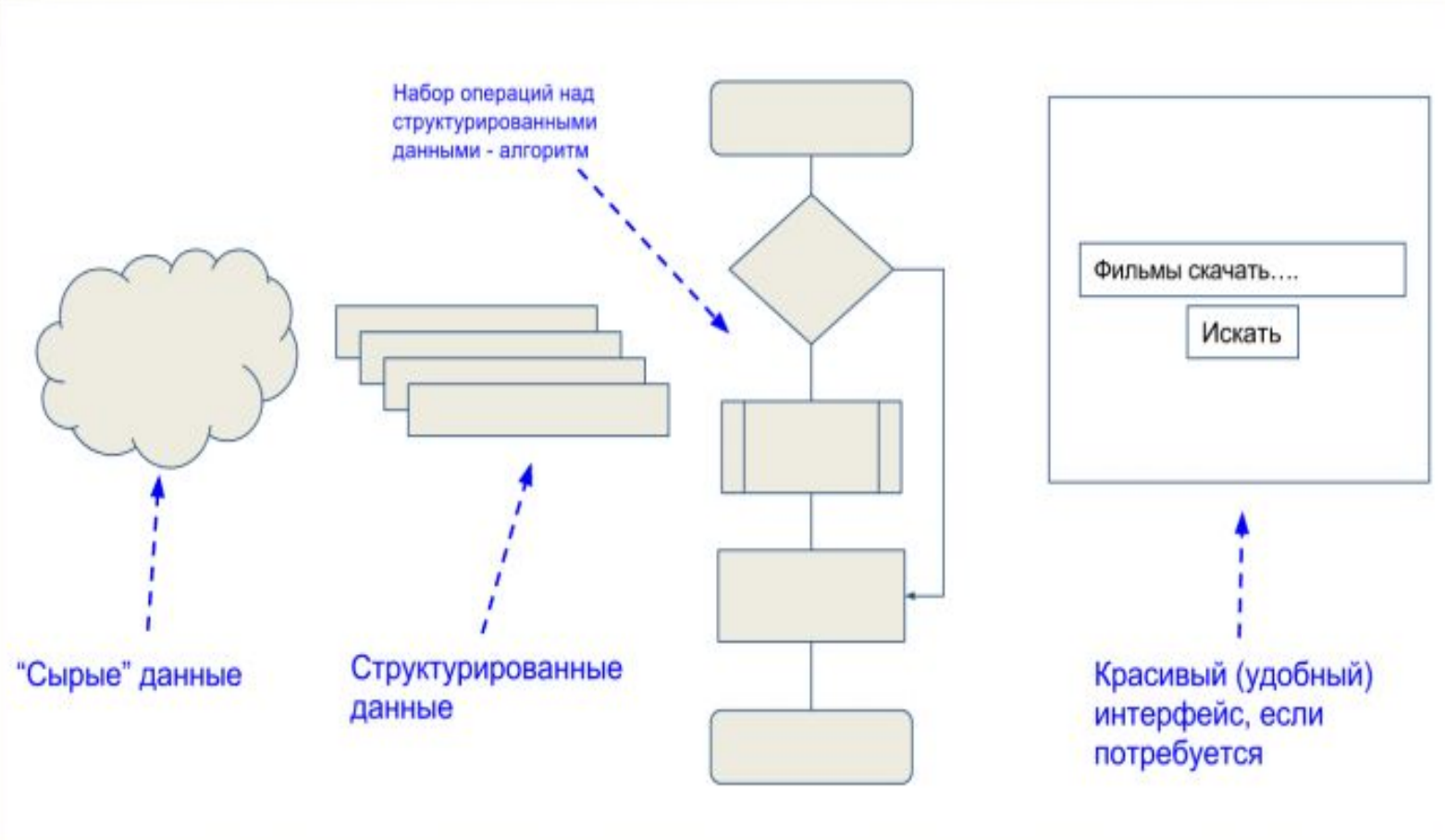
- **Компилятор**

Компилятор — программа, преобразующая текст, написанный на алгоритмическом языке, в программу, состоящую из машинных команд. Компилятор создаёт законченный вариант программы на машинном языке.

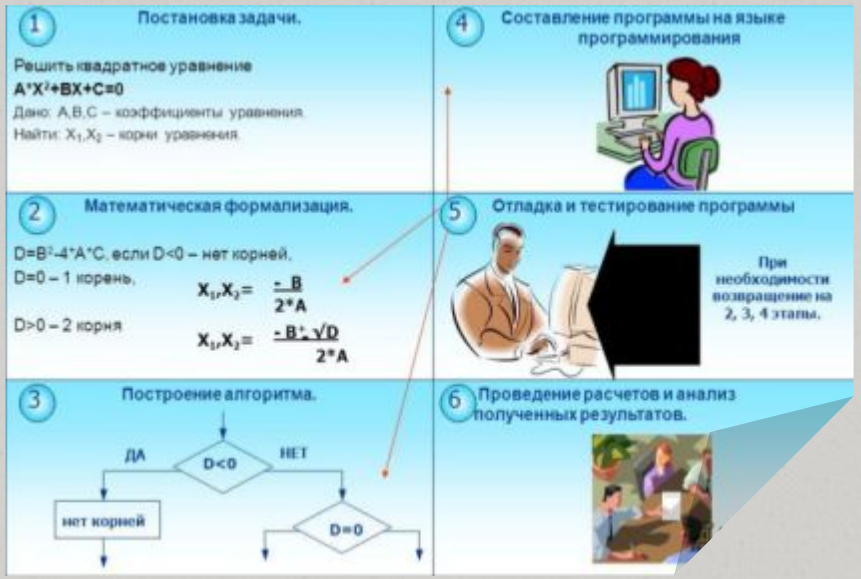
Понятия «алгоритм» и «программа».

Алгоритм решения задачи – точное описание порядка действий, которые надо выполнить для решения задачи.

«**Алгоритмы + структуры данных = программы**»



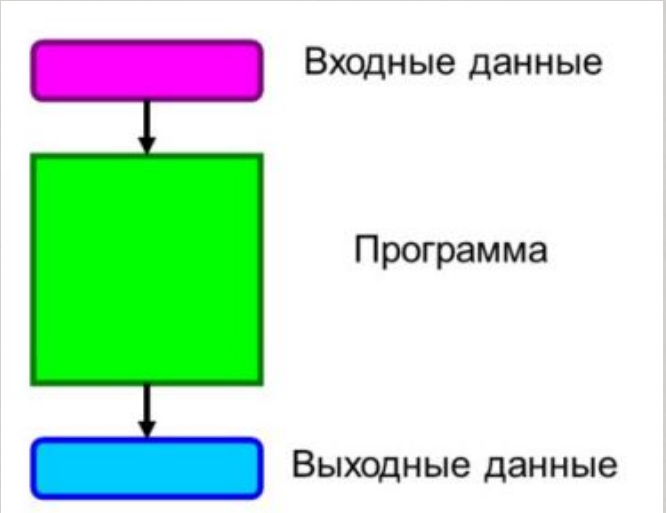
Этапы решения задачи на компьютере



1. Постановка задачи.
2. Формализация задачи.
3. Построение алгоритма.
4. Составление программы на языке программирования.
5. Отладка и тестирование программы.
6. Проведение расчетов и анализ полученных результатов.

Данные

- Совокупность величин, с которыми работает компьютер, принято называть данными.
- По отношению к программе данные делятся на исходные, результаты (окончательные данные) и промежуточные данные, которые получаются в процессе вычислений.
- Данные - это множество величин.



Величины

- Всякая величина занимает свое определенное место в памяти компьютера, а значение этой величины определяется двоичным кодом в этой ячейке.
- У всякой величины имеются три основных свойства: имя, значение и тип.
- Величины делятся на константы и переменные.

Способы записи алгоритмов

- Для записи алгоритмов используют самые разнообразные средства. Выбор средства определяется типом исполняемого алгоритма. Выделяют следующие основные способы записи алгоритмов:
- - **вербальный**, когда алгоритм описывается на человеческом языке;
- - **символьный**, когда алгоритм описывается с помощью набора символов;
- - **графический**, когда алгоритм описывается с помощью набора графических изображений.
- Общепринятыми способами записи являются графическая запись с помощью блок-схем и символьная запись с помощью какого-либо алгоритмического языка.
- Описание алгоритма с помощью блок схем осуществляется рисованием последовательности геометрических фигур, каждая из которых подразумевает выполнение определенного действия алгоритма. Порядок выполнения действий указывается стрелками. Написание алгоритмов с помощью блок-схем регламентируется ГОСТом. Внешний вид основных блоков, применяемых при написании блок схем, приведен на рисунке:

Структура алгоритмов

- В 1969 году известным голландским ученым-программистом Э. В. Дейкстрой было доказано, что алгоритм для решения любой логической задачи можно составить только из структур **следование, ветвление, цикл.**
- Их называют базовыми алгоритмическими структурами.
- Методика программирования, основанная на этой теореме, называется **структурным программированием**.

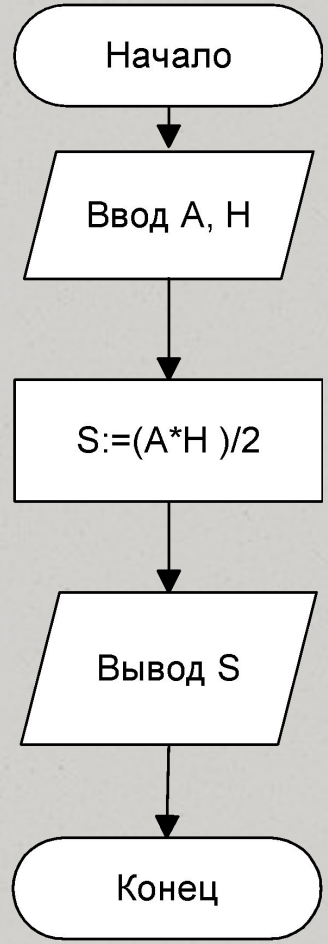
Линейный алгоритм

- **Следование** - алгоритмическая конструкция, отображающая последовательный порядок действий.
- Алгоритмы, в которых используется только структура «следование», называются **линейными** алгоритмами.

Пример линейной задачи

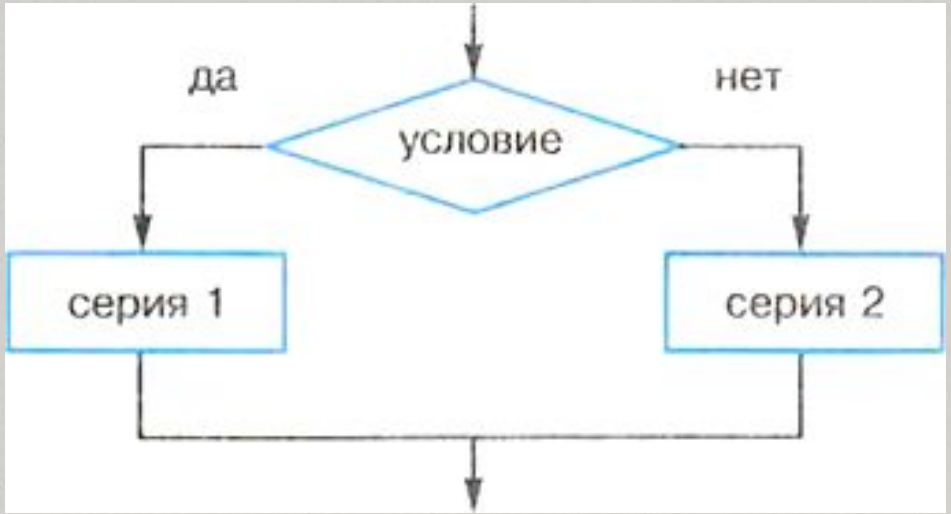
Задача №1. Найдите площадь треугольника с основанием A, высотой H.

алг
нач
S:=(A*H)/2
кон



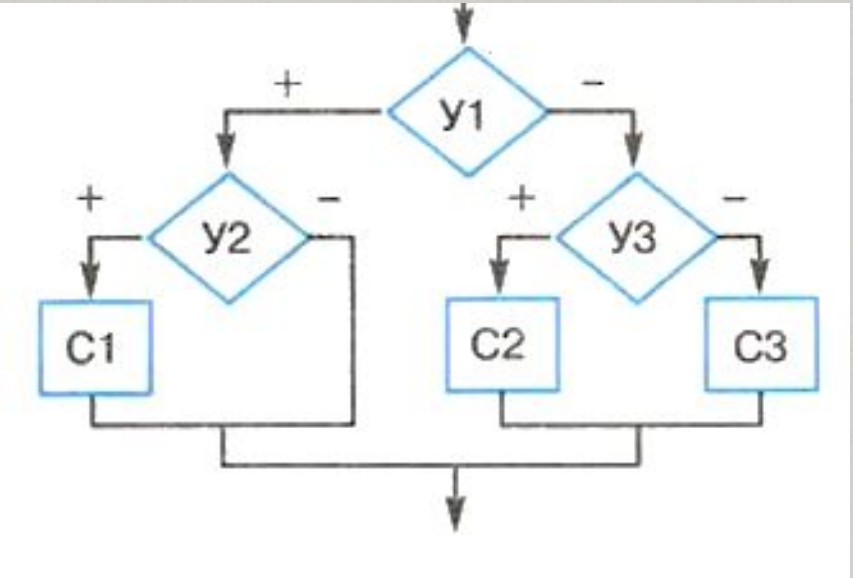
Ветвление

- Ветвление — алгоритмическая альтернатива.
- Управление передаётся одному из двух блоков в зависимости от истинности или ложности условия.
- Затем происходит выход на общее продолжение.



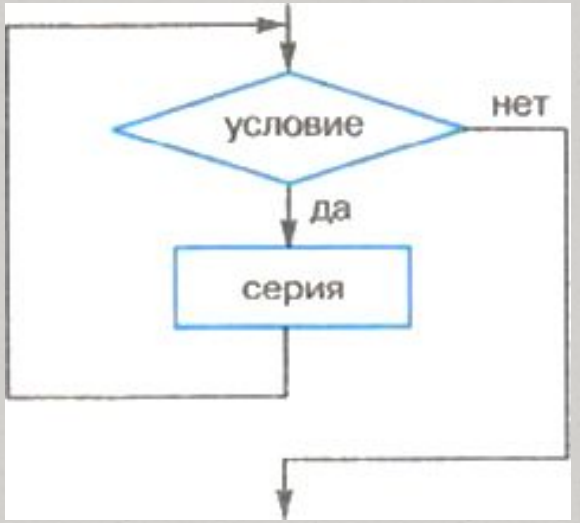
Ветвление

если <У1>
то если <У2>
то <С1>
все
иначе если <У3>
то <С2>
иначе <С3>
все
все



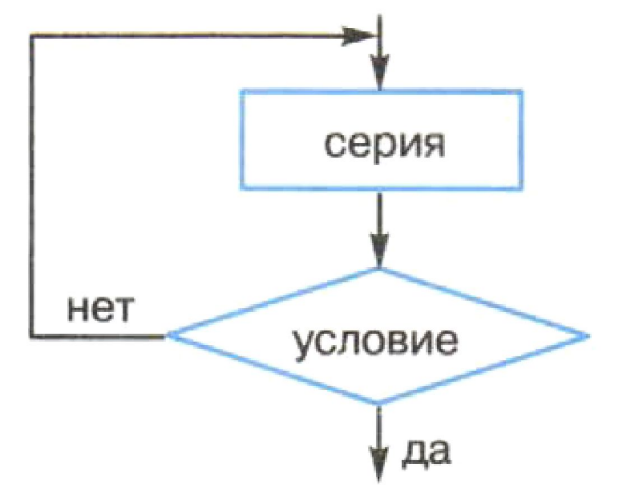
Цикл

- Цикл — повторение некоторой группы действий по условию.
- Различают два типа цикла.
- Первый — цикл с предусловием: **цикл-пока.**
- Пока условие истинно, выполняется серия, образующая тело цикла.



Цикл

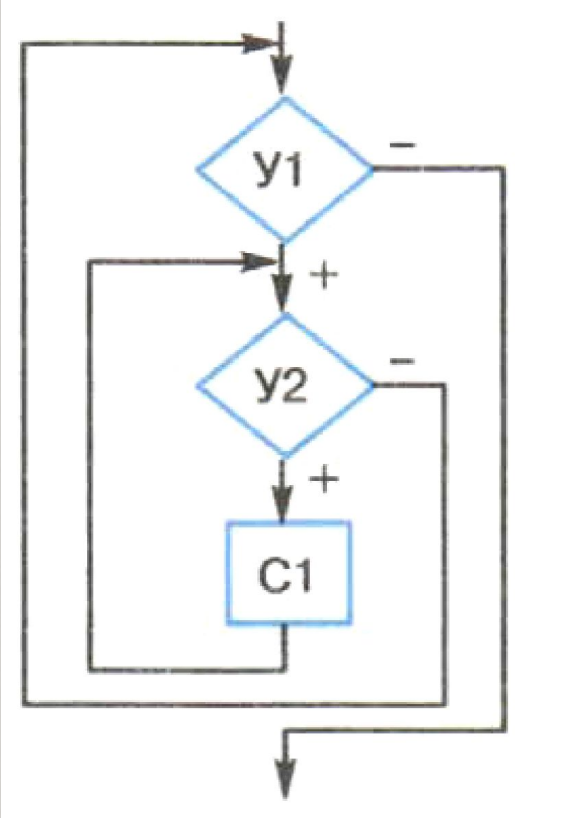
- Второй тип циклической структуры — цикл с постусловием: **цикл-до**.
- Здесь тело цикла предшествует условию цикла. Тело цикла повторяет свое выполнение, если условие ложно.
- Повторение прекращается, когда условие становится истинным.



Цикл

- Если блок, составляющий тело цикла, сам является циклической структурой, то имеют место вложенные циклы.
- Вложенная конструкция записывается смещенной по строке на несколько позиций вправо относительно внешней для нее конструкции

```
пока <У1>
нц
        пока <У2>
        нц
                <С1>
        кц
кц
```



Язык программирования - Python

Python - интерпретируемый, объектно-ориентированный, высокоуровневый язык программирования общего назначения. Синтаксис Python прост в изучении, в нем придается особое значение читаемости кода, а это сокращает затраты на сопровождение программных продуктов. Python поддерживает модули и пакеты, поощряя модульность повторное использование кода. Интерпретатор Python и большая стандартная библиотека доступны бесплатно в виде исходных и исполняемых кодов для всех основных платформ и могут свободно распространяться.

Особенности языка

- Минималистичность синтаксиса
- Простота изучения
- Широкие возможности
- Поддержка разных парадигм программирования
- Интерпретируемый
- Динамическая типизация
- Открытость
- Кроссплатформенность
- Существование альтернативных реализаций, которые решают определенные проблемы (StacklessPython, PyPy) или интегрируются его в определенную платформу (IronPython для .Net, Jython для Java)
- Активно развивается

История языка Python

- разрабатывался с конца 80-х, выпущен в феврале 1991 г.
- создание Python было начато Гвидо ван Россумом (тогда сотрудник голландского института CWI, сейчас – разработчик в Dropbox).
- назван в честь британского телешой «Летающий цирк Монти Пайтона».
- создавался под влиянием других языков, вобрал в себя множество их возможностей.
- на данный момент существуют две активные ветки языка - Python 2 и Python 3.

- **Высокоуровневый язык программирования** — язык программирования, разработанный для быстроты и удобства использования программистом. Основная черта высокоуровневых языков — это абстракция, то есть введение смысловых конструкций, кратко описывающих такие структуры данных и операции над ними, описания которых на машинном коде (или другом низкоуровневом языке программирования) очень длинны и сложны для понимания.

- Высокоуровневые языки программирования были разработаны для платформенной независимости сути алгоритмов. Зависимость от платформы перекладывается на инструментальные программы — трансляторы, компилирующие текст, написанный на языке высокого уровня, в элементарные машинные команды (инструкции). Поэтому, для каждой платформы разрабатывается платформенно-уникальный транслятор для каждого высокоуровневого языка, например, переводящий текст, написанный на Delphi в элементарные команды микропроцессоров семейства x86.

- Так, высокоуровневые языки стремятся не только облегчить решение сложных программных задач, но и упростить портирование программного обеспечения. Использование разнообразных трансляторов и интерпретаторов обеспечивает связь программ, написанных при помощи языков высокого уровня, с различными операционными системами программируемыми устройствами и оборудованием, и, в идеале, не требует модификации исходного кода (текста, написанного на высокоуровневом языке) для любой платформы.

Области применения

- Скрипты, утилиты
- Научная сфера
- Исследование данных
- Веб-приложения
- Сервисы
- Разработка игр

Сферы, в которых применить Python нельзя

- Создание драйверов устройств
- Низкоуровневое программирование

Преимущества языка

- Простота
- Читабельность
- Мощная стандартная библиотека и огромный выбор сторонних библиотек и модулей
- Краткость кода и экономия времени разработчика
- Возможность связывания с кодом на других языках

недостатки

- Низкая по сравнению с компилируемыми языками скорость работы
- GIL (Global Interpreter Lock)

Интерпретатор и интегрированная среда разработки

- Интерпретатор – программа (разновидность транслятора), выполняющая интерпретацию.

- Официальный сайт Python: https://python.org/
- Загрузить интерпретатор: https://www.python.org/downloads

Два режима работы:

- Выполнение программ;
- Интерактивный режим.

Интегрированная среда разработки (IDE) http://www.jetbrains.com/pycharm/

- Интегрированная среда разработки, IDE (Integrated development environment) – система программных средств, используемая программистами для разработки программного обеспечения (ПО).

Среда разработки обязательно включает в себя:

- Текстовый редактор,
- Компилятор и/или интерпретатор или средства интеграции с ним,
- Средства автоматизации сборки, отладчик.

Также IDE может включать:

- Средства интеграции с системами контроля версий,
- Инструменты конструирования графического интерфейса пользователя.

Контрольные вопросы

1. Что такое «алгоритм решения задачи»?
2. Какие вы знаете способы записи алгоритма?
3. В чем особенности алгоритма, который называют «программой»?
4. Почему языки программирования высокого уровня так называются?
5. Что такое «транслятор»? Какие функции он выполняет?
6. Какие виды трансляторов вы знаете? В чем особенность каждого вида?
7. Что включает в себя система программирования?

Линейный алгоритм. Задача 3. Построение алгоритма

Блюдо	Цена
Борщ	35
Котлета	40
Каша	20
Чай	3

Используя данные таблицы определить общую стоимость обеда в столовой. Определить, во сколько раз возрастёт стоимость обеда, если цена котлеты увеличится вдвое

Постановка задачи (формализованная): Имеется четыре числа, которые требуется просуммировать (обозначим их переменными a, b, c и d соответственно).

Сумму их значений обозначим S1. Требуется найти также величину S2=S1+b

и определить отношение S2/S1 (обозначим это отношение переменной res).

В результате нужно вывести значения переменных S1 и res .

Ветвление. Задача 2. Составить алгоритм решения задачи в виде Блок-схемы

- Построить алгоритм решения ввода значения температуры воздуха t и выдачи текста «Хорошая погода!», если t > 10 градусов и текста «Плохая погода!», если t <= 10 градусов.
- Постановка задачи: Исходными данными является значение t, необходимо сформировать строку s.
- При t < 10 s=’Плохая ␣ погода!’ , иначе s=’ Хорошая ␣ погода!’.

Ветвление. Задача 3. Составить алгоритм решения задачи в виде Блок-схемы

- Составить алгоритм ввода оценки P , полученной учащимся, и выдачи текста «Молодец!», если P = 5, «Хорошо!», если P = 4 и «Лентяй!», если P <= 3.