Как решать задачу на компьютере?

Этапы решения задач на компьютере включают: постановку задачи, формализацию (построение математической модели), разработку алгоритма, создание программы на языке программирования, тестирование и отладку, а также получение и анализ результатов. Каждый этап имеет свои задачи и представляет собой последовательный процесс, где результат одного этапа становится входными данными для следующего.

1. Постановка задачи

- **Что это**: Определение цели, исходных данных, формата и области значений результатов.
- **Цель**: Полностью и чётко сформулировать, что нужно решить, используя естественный язык.

2. Формализация задачи

- **Что это**: Создание математической модели, перевод условий задачи на язык математики.
- **Цель**: Установить математические соотношения между объектами задачи.

3. Построение алгоритма

- **Что это**: Разработка последовательности действий для решения задачи на основе математической модели.
- **Цель**: Создать пошаговое решение задачи, которое можно будет реализовать на компьютере.

4. Составление программы

- **Что это**: Перевод алгоритма на конкретный язык программирования (например, Паскаль).
- **Цель**: Написать код, который компьютер сможет исполнить.

5. Тестирование и отладка

- **Что это**: Проверка программы на правильность работы и исправление ошибок.
- **Цель**: Найти и устранить синтаксические и логические ошибки, чтобы программа работала корректно.

6. Получение и анализ результатов

- **Что это**: Выполнение программы с набором тестовых данных и анализ полученных результатов.
- **Цель**: Убедиться, что решение соответствует поставленной задаче и при необходимости вернуться к предыдущим этапам для его улучшения.

Подробнее с примером: https://stepik.org/lesson/347486/step/1

Там есть три вопроса и задача – сможете решить? Для этого зарегистрируйтесь – очень, очень полезно ☺

P.S. Слева в окне браузера увидите СОДЕРЖАНИЕ! Удобно)

Еще примеры ☺

Задача.

Вычисление индекса массы тела и его интерпретация

I этап. Неформальная постановка

Входные данные:

Описание	Пример	Недопустимые значения
Имя пользователя	Paul, Ольга, Артем	-
Возраст (количество полных лет)	23, 45, 16	<10
Рост (м)	1.78, 1.65, 1.85	0 < ИЛИ > 3
Вес (кг)	65, 58.7, 80.3, 75	<0 ИЛИ > 500

Выходные данные:

Описание	Пример
Индекс массы тела	18.45, 24.12, 27.1
Заключение	Недостаточная масса тела, Нормальная масса тела, Избыточная масса тела, Ожирение

Описание переменных:

Описание	Имя	Тип
Имя пользователя	**name**	**string**
Возраст (количество полных лет)	**age**	**int**
Рост (м)	**height**	**float**
Вес (кг)	**weight**	**float**
Индекс массы тела	**bmi**	**float**
Заключение	**description**	**string**

Используемые формулы:

Индекс массы тела рассчитывается по формуле:

$$bmi = m / h^2$$

где: *bmi*– индекс массы тела в кг/м²,
m – масса тела в килограммах,
h – рост в метрах.

Функциональные возможности:

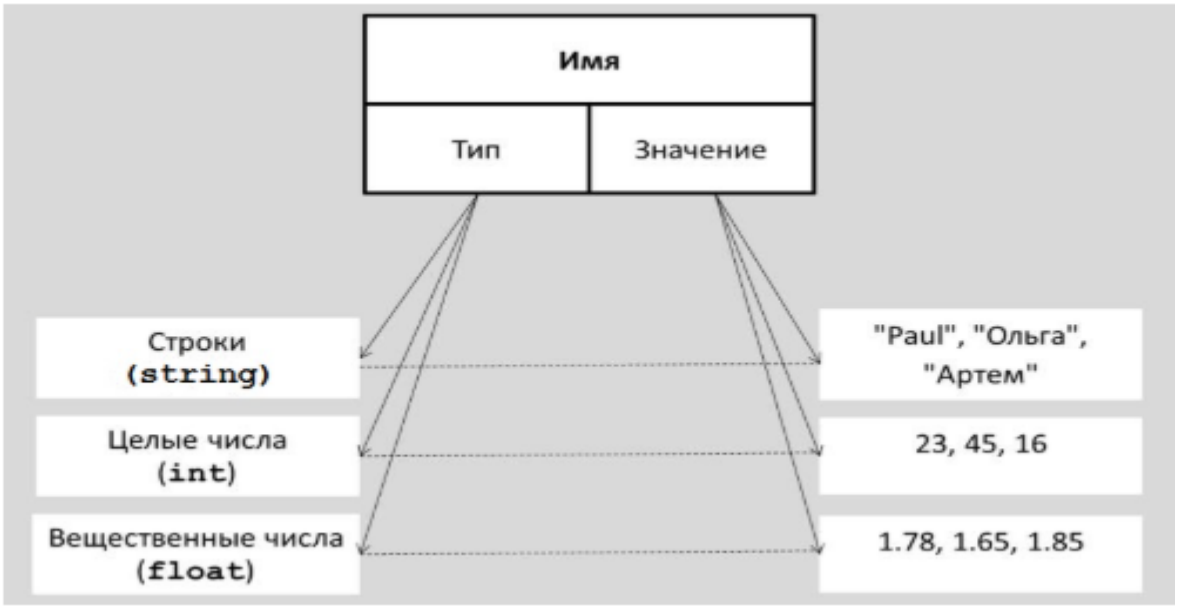
Создаваемая программа должна:

- спросить у пользователя его имя;
- спросить у пользователя, сколько полных лет ему исполнилось;
- вывести приветствие на экран;
- спросить у пользователя его рост и вес;
- вычислить индекс массы тела;
- вывести индекс массы тела на экран;
- вывести заключение по посчитанному индексу

Немного теории ;)

Переменные и типы

Для хранения данных в программе используются **переменные**. Простыми словами, переменная – это некоторый объект, который имеет имя, значение и тип:



В нашем случае для описания имени человека будет использоваться переменная строкового типа (**string**), и она может принимать текстовые значения, заключенные в двойные или одинарные кавычки. Для обозначения возраста используется целая переменная (тип **int**), в которую можно занести любые целые значения. Для описания веса и

роста будем использовать вещественный тип (**float**). При вводе значений этого типа в качестве разделителя целой и дробной части используется только **точка**.

Существует несколько **правил** для выбора имен переменных.

1. Имя должно состоять из букв, цифр и знаков подчеркивания, начинаться с буквы. Не может совпадать с ключевыми словами языка.
2. Python различает большие и маленькие буквы в имени (переменная **x** и **X** – разные переменные).
3. Имя переменной должно максимально точно соответствовать хранимым в ней данным.
4. Имена переменных могут состоять из нескольких слов, тогда они пишутся через подчеркивание **вот_так**.

Примеры: **age, user_age, weight, weight_person**

Немного теории:

- *Ввод и вывод*
- *Оператор присваивания*
- *Условный оператор*
- *Вложенные конструкции*
- *Встроенные функции Python*

Ввод и вывод

Для **ввода данных** используется оператор **input()**, который считывает строку текста, введенную пользователем в консоли, и заносит ее в переменную. Следующий оператор вводит строку текста и заносит ее в переменную name.

```
name = input()
```

Для **вывода значения переменной** используется оператор **print(),** оператор выведет на экран значение переменной **name**:

```
print(name)
```

С помощью оператора **print()** можно выводить строки в кавычках и значения переменных одновременно, при выводе они разделяются запятой.

```
print("Привет, ", name, "!")
```

Для **ввода числовых значений** в переменные необходимо перед оператором ввода **input()** указать тип значения (**int** или **float**). Следующий оператор вводит целое значение в переменную **age**:

```
age = int(input())
```

Кроме того Python допускает использование подсказки пользователю непосредственно в операторе **input()**. При использовании такой подсказки ввод данных пользователем при

выполнении программы будет осуществляться сразу после текста подсказки на той же строке. Следующий оператор вводит целое значение в переменную **age**, выдавая пользователю подсказку:

```
age = int(input("Сколько Вам полных лет? "))
```

Оператор присваивания

Вычисление выражения и занесение его значения в некоторую переменную осуществляется с помощью **оператора присваивания**. Например, следующий оператор вычисляет индекс массы тела и результат заносит в переменную **bmi**:

```
bmi = weight / height ** 2
```

В выражениях допустимо использовать **арифметические операции** (сложение «**+**», вычитание «**-**», умножение «*****», деление «**/**», целочисленное деление «**//**», остаток от деления «**%**», возведение в степень «******»). Python поддерживает привычный порядок выполнения операций: сначала возведение в степень, затем умножение, деление, целочисленное деление и остаток (все они имеют одинаковый приоритет, применяются последовательно слева направо), последним выполняется сложение или вычитание. Для изменения приоритета используются круглые скобки.

Примеры выражений:

- *3 + 2 * 5 ** 2 - 1= 52*
- *(3 + 2) * 5 ** 2 - 1 = 124*
- *5 ** 2 / 2 = 12.5*
- *63 % 10 = 3*
- *63 % 10 + 4 = 7*
- *63 % (10 + 4) = 7*
- *63 // 10 = 6*
- *63 // 10 + 4 = 10*
- *63 // (10 + 4) = 4*

Условный оператор

Для реализации возможности выполнения различных действий в зависимости от условия в Python используется **условный оператор if…elif…else**, синтаксис (правила записи) которого следующий (разделы **elif** и **else** могут отсутствовать):

```
if условие:
    операторы
elif  условие:
    операторы
...
else:
    операторы
```

где:

условие – логическое выражение, которое может включать **знаки отношений**

- (равно «**==**» (два подряд равенства),
- не равно «**!=**»,
- больше «**>**»,
- меньше «**<**»,
- больше или равно«**>=**»,
- меньше или равно «**<=**»),
- **логические операции** (И «**and**», ИЛИ «**or**», НЕ «**not**»);

операторы – любые операторы, допустимые в языке Python.

Результатом вычисления логических выражений является либо Истина (***True***), другими словами, выражение «верно», либо Ложь (***False***) – выражение «неверно». При использовании логической операции И (**and**) выражение верно, если все его элементы верны. При использовании логической операции ИЛИ (**or**) выражение верно, если хотя бы один его элемент верен.

Примеры:

- 3 > 5 результат **False**
- 3 == (1 + 2) результат **True**
- 3 >= (1 + 2) результат **True**
- 3 != (1 + 2) результат **False**
- 3 < 5 **and** 3 >= 4 результат **False**
- 3 < 5 **or** 3 >= 4 результат **True**
- 3 < 5 **and** not (3 >= 4) результат **True**
- 3 > 5 **or** 3 >= 4 результат **False**

Вложенные конструкции

После условия в операторах **if** и **elif,** а также после **else** ОБЯЗАТЕЛЬНО ставится знак двоеточия «**:**», который означает наличие **вложенных конструкций**.

Вложенные инструкции в Python записываются в соответствии с одним и тем же шаблоном, когда основная инструкция завершается двоеточием, вслед за которым располагается вложенный блок кода, с отступом под строкой основной конструкции.

Вложенные инструкции объединяются в блоки по величине отступов. Отступ может быть любым, главное, чтобы в пределах одного вложенного блока отступ был одинаков (обычно это 4 пробела).

В примере
Оператор 1 и **Оператор 2** относятся к вложенному блоку **Основной конструкции**, а **Оператор 3** находится на уровне **Основной конструкции.**

```
Основная конструкция:
    Оператор 1
    Оператор 2
Оператор 3
```

Встроенные функции Python

В языке Python реализовано много встроенных функций, использующихся для различных целей, вот некоторые из них:

- **abs(x)** – вычисляет модуль числа x;
- **round(x, n)** - округляет число **x** до **n** знаков после запятой;
- **min(x, y, z)** – находит минимальное значение среди перечисленных через запятую элементов;
- **max(x, y, z)** – находит максимальное значение среди перечисленных через запятую элементов.

Примеры:

abs(*-4****)*** *= 4*

abs(*5****)*** *= 5*

round(*3.141569****)*** *= 3*

round(*3.141569, 4****)*** *=3.1416*

min(*3, 4, 6.8, -3, -1.89****)*** *= -3*

max(*3, 4, 6.8, -3, -1.89****)*** *= 6.8*

Пояснение. В Python для округления чисел используется банковский способ округления, который отличается от математического только, если в следующей позиции после позиции округления стоит 5. При математическом округлении в этом случае число округляется в большую по модулю сторону. При банковском - к ближайшему чётному.

Например:

round*(2.65, 1) = 2.6*

round*(2.75, 1) = 2.8*

II этап. Реализация задачи

1. Написать фрагмент программы, которая спрашивает у пользователя его имя .

Код:

```
print("Ваше имя?")
name = input()
print("Привет, ", name, "!")
```

2. Дописать программу так, чтобы пользователь, кроме имени, вводил свой возраст, рост и вес.

Код:

```
age = int(input("Сколько Вам полных лет? "))
height = float(input("Ваш рост? "))
weight = float(input("Ваш вес? "))
```

Результат:

3. Вычислить индекс массы тела и вывести его значение на экран.

Код

```
bmi = weight / height ** 2
print("Ваш индекс массы тела: ", bmi)
```

Результат

4. Округлить результат до 2-х знаков после запятой (количество знаков может быть любым, в таблице для интерпретации результатов нашей задачи используется вещественные числа с двумя знаками после запятой, значит более точные результаты не нужны).

Код:

```
bmi = weight / height ** 2
print("Ваш индекс массы тела: ",round(bmi, 2))
```

Результат:

Но необходимо отметить, что в программе будет храниться неокругленное значение переменной bmi, оно будет округляться только для вывода. Чтобы в программе можно было использовать значение с 2 знаками после запятой, необходимо округлить значение переменной после вычисления (результат будет такой же как при округлении при выводе).

Код:

```
bmi = weight / height ** 2
bmi = round(bmi, 2)
print("Ваш индекс массы тела: ", bmi)
```

5. Объяснить пользователю результат вычисления в соответствии с таблицей Всемирной Организации Здравоохранения. При этом на экран предполагается вывести

сообщение типа «Вы относитесь к группе с нормальной массой тела», поэтому в переменную description словосочетание занесено в творительном падеже.

Если индекс массы тела меньше 18.5
description = "недостаточной массой тела"
иначе если индекс массы тела меньше 25
description = "нормальной массой тела"
иначе если индекс массы тела меньше 30
description = "избыточной массой тела"
иначе
description = "ожирением"

Код:

```
if bmi < 18.5:
  description = "недостаточной массой тела."
elif bmi < 25:
  description = "нормальной массой тела."
elif bmi < 30:
  description = "избыточной массой тела."
else:
  description = "ожирением."
```

Результат для различных входных данных:

6. Реализовать проверку входных данных. При вводе неверных данных получается совершенно неадекватный результат (см. предыдущий пункт). Поэтому прежде чем что-то сделать в программе, необходимо проверить вводимые данные, если они неверные, то вывести сообщение пользователю. Программа должна работать, только если ввод пользователя верный.

Тест – это набор данных, который может поступить на вход программы.

Ошибочные данные:

- неверный возраст (например, < 10)
- неверный рост (< 0 or > 3)
- неверный вес (< 0 or > 500)

В общем виде структуру программы с проверкой входных данных можно описать в следующим образом:

ввод данных
if входные данные принимают недопустимые значения:
print("Ошибочные входные данные")
else:
операторы программы
вывод результатов

Код полной программы:

(можно копировать и проверить; не пугайтесь длинных имен переменных)

```
print("Ваше имя?")
name = input()
print("Привет, ", name, "!")
age = int(input("Сколько Вам полных лет? "))
height = float(input("Ваш рост? "))
weight = float(input("Ваш вес? "))
if age < 10 or height <= 0 or height > 3 or weight <= 0 or weight > 500:

    print("Ошибочные входные данные")
else:
    bmi = weight / height ** 2
    bmi = round(bmi, 2)
    print("Ваш индекс массы тела: ", str(bmi))

    if bmi < 18.5:
       description = "недостаточной массой тела."

    elif bmi < 25:
        description = "нормальной массой тела."

    elif bmi < 30:
        description = "избыточной массой тела."
    else:

        description = "ожирением."

    print("Вы относитесь к группе людей с", description)
```

Этап III. Отладка и тестирование.

На этом этапе необходимо проверить работоспособность программы для различных входных данных. Естественно, невозможно перебрать все возможные варианты ввода пользователя, но необходимо их систематизировать и выделить те, которые позволяют проверить все вычисления и условные переходы.

Описание процесса тестирования можно оформить в виде таблицы. В ней привести вариант входных данных, правильный результат, который должна выдать программа, а также предусмотреть поле, в котором отметить совпадает ли реальный результат с верным.

Набор входных данных	Правильный результат	Верно?
age : -4; height :1.5; weight : 6	Ошибочные входные данные	✓
age : 9; height :-1.5; weight : 65	Ошибочные входные данные	✓
age : 25; height :3.8; weight : 60	Ошибочные входные данные	✓
age : 40; height :1.8; weight : -5	Ошибочные входные данные	✓
age : 26; height :1.8; weight : 900	Ошибочные входные данные	✓
age : 30; height :1.8; weight : 90	Ваш индекс массы тела: 27.78 Вы относитесь к группе людей с избыточной массой тела.	✓
age : 25; height :1.6; weight : 45	Ваш индекс массы тела: 17.58 Вы относитесь к группе людей с недостаточной массой тела.	✓
age : 30; height :1.7; weight : 63	Ваш индекс массы тела: 21.8 Вы относитесь к группе людей с нормальной массой тела.	✓
age : 16; height :1.7; weight : 162	Ваш индекс массы тела: 56.06 Вы относитесь к группе людей с ожирением.	✓

Задача

Вычислить индекс массы тела в зависимости от роста, веса и возраста, а затем проинтерпретировать результат в соответствии с рекомендациями Всемирной Организации Здравоохранения:

Индекс массы тела, возраст < 45	Индекс массы тела, возраст >= 45	Описание
меньше 18,5	меньше 22	Недостаточная масса тела
18,5 - 24,99	22 - 26,99	Нормальная масса тела
25 – 29.99	27– 31.99	Избыточная масса тела
больше или равно 30	больше или равно 32	Ожирение

Пояснение.

Эту задачу будем решать на основе программы, созданной на предыдущих шагах.

Код исправленной программы:

```
age = int(input())
height = float(input())
weight = float(input())
if age < 10 or height <= 0 or height > 3 or weight <= 0 or weight > 500:
    print("Ошибочные входные данные")
else:
    bmi = weight / height ** 2
    bmi = round(bmi, 2)

    if bmi < 18.5:
        description = "недостаточной массой тела."
    elif bmi < 25:
        description = "нормальной массой тела."
    elif bmi < 30:
        description = "избыточной массой тела."
    else:
        description = "ожирением."

    print("bmi=", bmi, "Вы относитесь к группе людей с", description)
```

Для решения поставленной задачи добавьте в эту программу операторы, которые позволяют учесть возраст при интерпретации результатов.

Sample Input 1:

35

1.8

120.71

Sample Output 1:

bmi= 37.26 Вы относитесь к группе людей с ожирением.

Sample Input 2:

18

1.68

82.79

Sample Output 2:

bmi= 29.33 Вы относитесь к группе людей с избыточной массой тела.

Sample Input 3:

50

1.97

89.67

Sample Output 3:

bmi= 23.11 Вы относитесь к группе людей с нормальной массой тела.

СЛАБО самому решить?

☺

```
age = int(input())
height = float(input())
weight = float(input())
if age < 10 or height <= 0 or height > 3 or weight <= 0 or weight > 500:
    print("Ошибочные входные данные")
else:
    bmi = weight / height ** 2
    bmi = round(bmi, 2)
    if (bmi < 18.5 and age < 45) or (bmi < 22 and age >= 45):
        description = "недостаточной массой тела."
    elif (bmi < 25 and age < 45) or (bmi < 27 and age >= 45):
        description = "нормальной массой тела."
    elif (bmi < 30 and age < 45) or (bmi < 32 and age >= 45):
        description = "избыточной массой тела."
    else:
        description = "ожирением."
    print("bmi=", bmi, "Вы относитесь к группе людей с", description)
```