**МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ,
СВЯЗИ И МАССОВЫХ КОММУНИКАЦИЙ РОССИЙСКОЙ ФЕДЕРАЦИИ**

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ТЕЛЕКОММУНИКАЦИЙ ИМ. ПРОФ. М.А. БОНЧ-БРУЕВИЧА»
(СПбГУТ)**

Санкт-Петербургский колледж телекоммуникаций им. Э.Т. Кренкеля

СПбГУТ))) Документ подписан простой электронной подписью

Сертификат: 00fd759708ffd39703
Владелец: Бачевский Сергей Викторович
Действителен с 15.03.2021 по 14.03.2026

УТВЕРЖДАЮ
Заместитель директора
по учебной работе
_____ О.В. Колбанева
21 апреля 2021 г

МЕТОДИЧЕСКИЕ УКАЗАНИЯ ПО ВЫПОЛНЕНИЮ ПРАКТИЧЕСКИХ РАБОТ

по учебной дисциплине
ОУД.09. ИНФОРМАТИКА

по специальности
10.02.04 Обеспечение информационной безопасности телекоммуникационных систем

среднего профессионального образования

Санкт-Петербург
2021

ОУД.09 Информатика. Методические указания по выполнению практических работ.

Составил Кривоносова Н.В. – Санкт-Петербург, 2021.

Методические указания содержат описания практических занятий, предусмотренных рабочей программой ОУД.09 Информатика. Каждая работа рассчитана на 2 академических часа, общий объём составляет 16 часов.

Методические указания предназначены для обучающихся очной формы обучения по специальности 10.02.04 Обеспечение информационной безопасности телекоммуникационных систем.

Рассмотрено и одобрено предметной (цикловой) комиссией информатики и программирования в компьютерных системах Санкт-Петербургского колледжа телекоммуникаций им. Э.Т. Кренкеля.

СОДЕРЖАНИЕ

Практическая работа 1
ПРОГРАММИРОВАНИЕ ЛИНЕЙНЫХ АЛГОРИТМОВ

1. **Цель работы**: ознакомиться с работой линейных алгоритмов в Python
2. **Задачи работы:**
   - уметь создавать информационные объекты сложной структуры
   - повторить принципы линейных алгоритмов
   - ознакомиться с основными принципами реализации линейных алгоритмов в Python

В соответствии общеобразовательной учебной дисциплиной «Информатика» в результате выполнения заданий ЛР, студент должен:

иметь представление:
- о методах поиска информации;
- о принципах кодирования информации;
- о системах счисления;

знать:
- различные подходы к определению понятия «информация»;
- методы измерения количества информации: вероятностный и алфавитный. Знать единицы измерения информации;
- использование алгоритма как способа автоматизации деятельности;

уметь:
- оценивать достоверность информации, сопоставляя различные источники;
- использовать готовые информационные модели, оценивать их соответствие реальному объекту и целям моделирования;
- осуществлять выбор способа представления информации в соответствии с поставленной задачей;
- представлять числовую информацию различными способами (таблица, массив, график, диаграмма и пр.);

3. **Подготовка к работе**

Ознакомиться с теоретической справкой

4. **Задание**
   1) Ознакомиться с теоретической справкой
   2) Выполнить задание 1 из п5
   3) Выполнить задание 2 из п5
   4) Выполнить задание 3 из п5
5. **Порядок выполнения работы**

Задание 1 сложности:

1. а) В среде Python открыть новый файл для создания программы командой Ctrl + N

б) Набрать в редакторе системы Python следующую программу:

```
a = int(input(“Введите a”))
b = int(input(“Введите b”))
c = int(input(“Введите c”))
S=a*b*c
print(S)
```

в) Запустить данную программу на выполнение и проверить правильность её работы для чисел 2, 4 и 6.

г) Запустить данную программу на выполнение и проверить правильность её работы для чисел 1, 0 и -1.

д) Запустить данную программу на выполнение и проверить правильность её работы для чисел -2, 3 и 10.

2. Написать программу, которая присваивает целой переменной A значение 10 и выводит это значение на экран.

3. Написать программу, которая запрашивает ввод целого числа в переменную B и выводит это число на экран. Проверить правильность работы программы на числах 1, -5, 256, 10455.

4. Написать программу, которая запрашивает ввод вещественного числа в переменную C, умножает это число на 2 и выводит результат на экран. Проверить правильность работы программы на числах 2.5, -7.33, 0, 782.234.
5. Написать программу для ввода значения величины X целого типа, присваивания величине Y действительного типа значения 5.5, вычисления значения величины Z = X - Y и вывода значения величины Z. Протестировать программу для X=5.5, X=0, X=-10.2
6. Написать программу для ввода значения величины X целого типа, присваивания величине Y действительного типа значения 2.5 , вычисления значения величины Z=X/Y и вывода значения величины Z. Протестировать программу для X=5, X=0, X=-8.75

Задание 2 сложности:

1. Написать на языке Python программу ввода четырёх целых чисел и вычисления их среднего арифметического. Протестировать программу на различных исходных данных (включая вещественные числа) и доказать правильность её работы.
2. Вводятся величины X,Y целого типа. Написать программу для обмена значений величин. Необходимо использовать вспомогательную величину T. Протестировать программу для X=5 и Y=-11.
3. Написать программу для вычисления дискриминанта d квадратного уравнения ax2 + bx + c = 0. Разработать тесты проверки правильности работы программы для вариантов, когда d0, d=0 и d.
4. Из железной полосы длиной L метров нужно изготовить обруч. На соединение концов уходит D метров полосы. Написать программу для вычисления радиуса R обруча. Протестировать программу для а) L=5.8, D=0.2, б) L=3.25, D=0.1
5. Найти площадь кольца, внешний радиус которого равен R1, а внутренний – R2 (R1R2). Протестировать программу для R1=5.6 и R2=3.8. Проверить ответ на калькуляторе.
6. Написать на языке Python программу для вычисления выражения:

S= (2x+y)(x-y)

Протестировать её для следующих исходных данных:

1) x=2, y=1 2) x=3, y=0 3) x=0, y=-2

Задание 3 сложности:

1. Заданы величины X,Y действительного типа. Написать программу для обмена значений величин. Использовать вспомогательные величины нельзя. Протестировать программу для X=-3 и Y=8.
2. Дано натуральное число X. Вычислить Y = X5. Разрешается использовать только три операции умножения. Протестировать программу для X=-2 и X=3.
3. Дано натуральное число X. Вычислить Y = 1 - 2X + 3X2 - 4X3. Разрешается использовать не более 8 арифметических операций. Допустимы: операции сложение, вычитание, умножение. Протестировать программу для X=0, X=1, X=-2.
4. Вычислить расстояние между двумя точками с координатами (X1,Y1) и (X2,Y2). Доказать правильность работы программы на трёх различных тестах.

6. Содержание отчёта

1. Название, цель работы
2. Выполнение п.5 (создание архивов)
3. Выводы по работе.

Приложение

Краткие сведения из теории

Линейное программирование – это набор методов, используемых в математическом программировании, также называемых математической оптимизацией. Эти методы используются для решения систем линейных уравнений и неравенств, перед которыми стоит цель максимизации или минимизации некоторой линейной функции. Линейное программирование используется в научных вычислениях, экономике, технических науках, производстве, транспорте, военном деле, логистике, энергетике и т. д.

Экосистема Python включает несколько мощных инструментов линейного программирования.

Линейное программирование – это набор математических и вычислительных инструментов, позволяющих найти конкретное решение системы, которое соответствует максимуму или минимуму какой-либо другой линейной функции. Линейное программирование – это фундаментальный метод оптимизации, десятилетиями применяемый в областях, требующих большого объема математических вычислений. Эти методы точны, сравнительно быстры и подходят для множества практических приложений.

Смешанно-целочисленное линейное программирование – это вид линейного программирования, которое фокусируется на обработке задач, где хотя бы одна переменная принимает дискретные целые, а не непрерывно меняющиеся значения.

Целочисленные переменные важны для правильного представления количеств, естественным образом выражаемых целыми числами, таких как число выпущенных самолетов или количество обслуженных клиентов.

Особенно важным видом целочисленных переменных являются бинарные переменные, имеющие лишь значения 0 или 1, и полезные при принятии решений вида «да»/«нет». Например, следует ли строить завод, включить или выключить машину. Также их можно использовать для имитации логических ограничений.

Смешанно-целочисленное линейное программирование позволяет преодолеть многие ограничения линейного программирования. Можно аппроксимировать нелинейные функции кусочно-линейными, использовать полунепрерывные переменные, логические ограничения модели. Это требовательный к ресурсам инструмент, но достижения в области компьютерного оборудования и программного обеспечения сделали его более доступным.

Линейное программирование на Python

Базовый метод решения задач линейного программирования называется симплекс-методом, другой популярный подход – метод внутренней точки. Задачи смешанного целочисленного линейного программирования решаются с помощью более сложных и ресурсоемких методов, таких как метод ветвей и границ.

Заметим, что почти все широко используемые библиотеки линейного программирования и смешанно-целочисленного линейного программирования написаны на языках Fortran, C или C++, так как линейное программирование требует интенсивной вычислительной работы с матрицами, часто очень большими. Соответствующие инструменты Python – это просто удобные интерфейсы для работы с низкоуровневыми библиотеками – солверами.

Для определения и решения задач линейного программирования мы будем использовать Python-библиотеки SciPy и PuLP.

Пример задачи линейного программирования

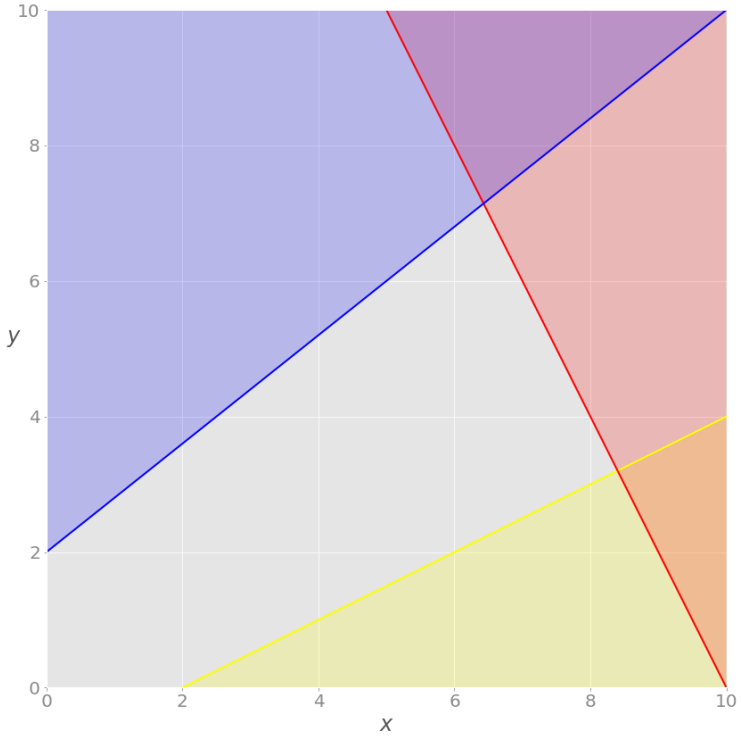
Рассмотрим следующую задачу максимизации:

$$\begin{aligned} \text{maximize} \quad & z = x + 2y \\ \text{subject to:} \quad & 2x + y \leq 20 \\ & -4x + 5y \leq 10 \\ & -x + 2y \geq -2 \\ & x \geq 0 \\ & y \geq 0 \end{aligned}$$

Нам нужно найти такие x и y, чтобы выполнялись «красное», «синее» и «желтое» неравенства, а также ограничения $x \geq 0$ и $y \geq 0$. При этом решение должно соответствовать максимально возможному значению z.

Независимые переменные, которые нужно найти (x и y) называют переменными решения (decision variables). Функция, которую необходимо максимизировать или минимизировать (z), – это целевая функция (objective function), функция стоимости (cost function) или просто цель (goal). Неравенства (или уравнения), которым необходимо удовлетворять, называются ограничениями (inequality constraints или equality constraints для обычных уравнений).

Проблему можно визуализировать следующим образом.



Красная линия представляет функцию$2x + y = 20$, а красная область над ней показывает, где красное неравенство не выполняется. Аналогично синяя линия – это $-4x + 5y = 10$, желтая линия – это$-x + 2y = -2$, окрашенные области – та часть плоскости, где неравенство не выполняется.

Каждая точка серой области удовлетворяет всем ограничениям и является потенциальным решением задачи. Эта область называется областью допустимых решений (feasible region), а ее точки – допустимыми решениями (feasible solutions).

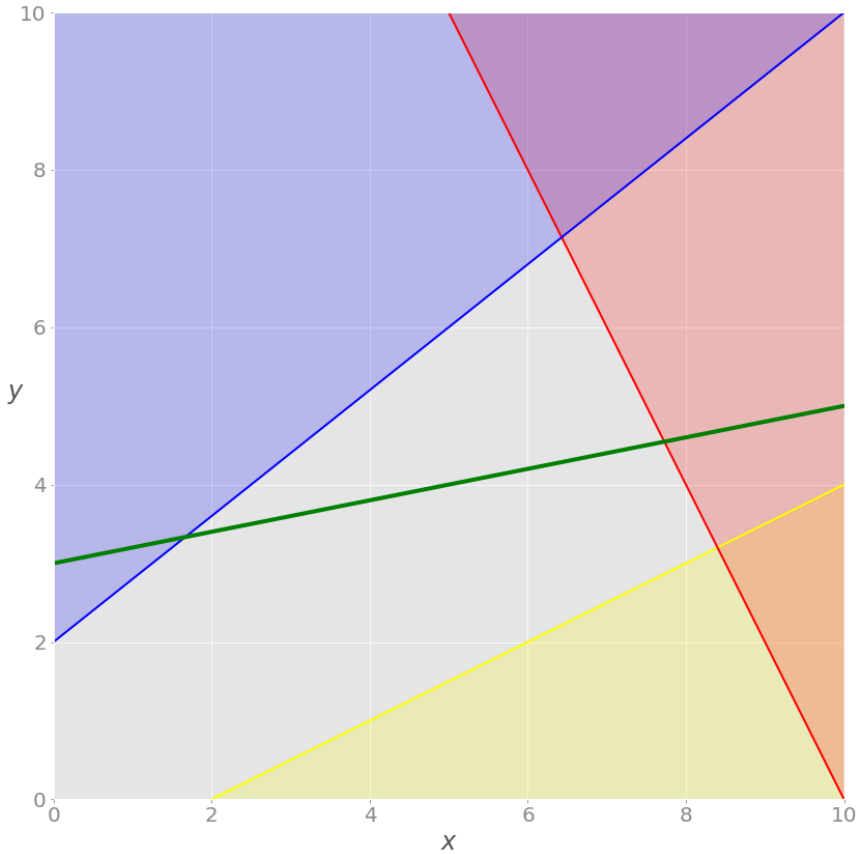
Мы хотим максимизировать z. Решение, соответствующее максимальному значению z, называют оптимальным решением.

Обратите внимание, что функция z линейна. Оптимальное решение должно находиться в одной из вершин области допустимых решений. Иногда весь край допустимой области или даже вся область может соответствовать одному и тому же значению z.

Представим, что в задачу введено дополнительное ограничение в виде равенства, окрашенного зеленым:

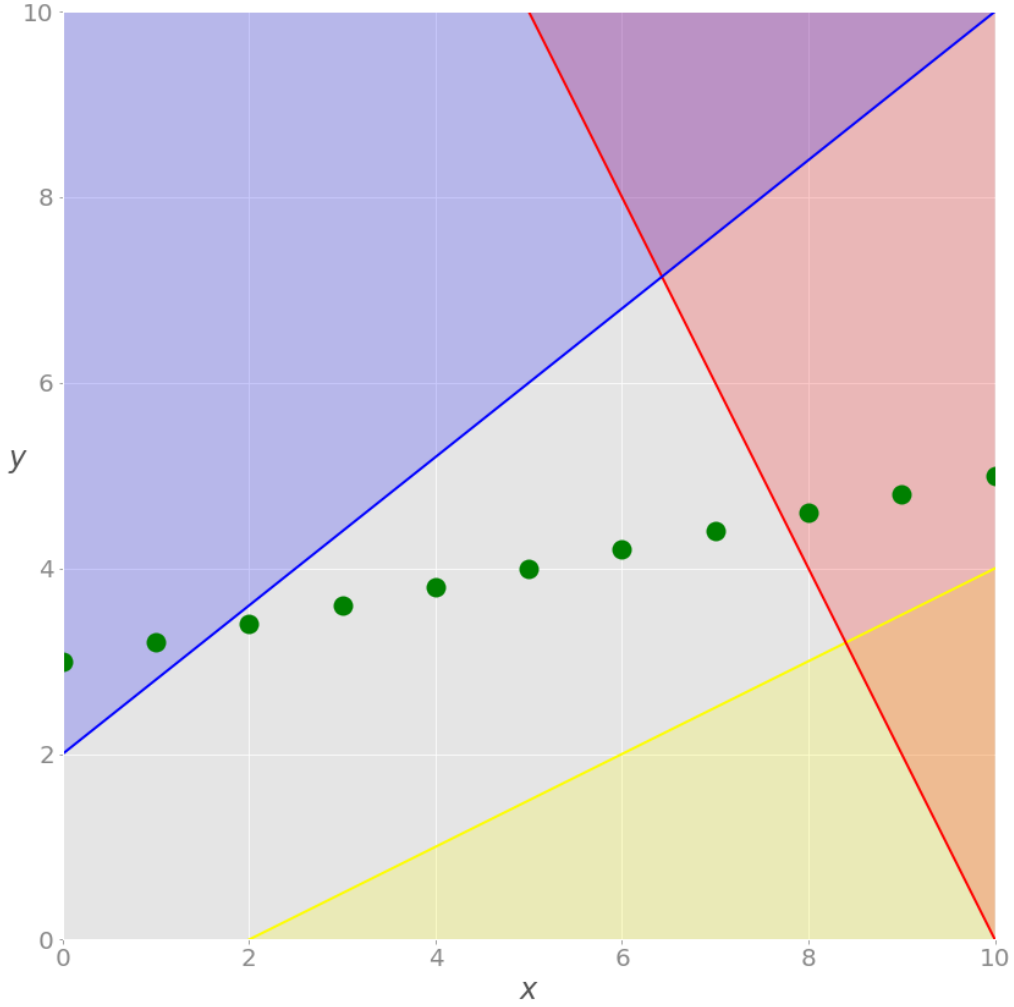
$$\begin{aligned} \text{maximize} \quad & z = x + 2y \\ \text{subject to:} \quad & 2x + y \leq 20 \\ & -4x + 5y \leq 10 \\ & -x + 2y \geq -2 \\ & -x + 5y = 15 \\ & x \geq 0 \\ & y \geq 0 \end{aligned}$$

Его можно визуализировать, добавив соответствующую зеленую прямую:



Теперь область допустимых решений не соответствует всей серой зоне. Это лишь часть зеленой линии, проходящей через серую область от точки пересечения с синей линией до точки пересечения с красной.

Если добавить требование, что все значения x должны быть целыми числами, то мы получим задачу смешанно-целочисленного линейного программирования, и набор возможных решений снова изменится:



Больше нет зеленой линии – только дискретные точки, где значение x является целым числом. Возможные решения – это зеленые точки на сером фоне.

Эти три примера иллюстрируют задачи линейного программирования – они имеют ограниченные допустимые области решений и конечные решения.

Когда ни одно решение не может удовлетворить все ограничения сразу, задача в рамках линейного программирования неразрешима.

Линейное программирование на Python. Практическая реализация

В этом руководстве мы будем использовать для решения описанной выше задачи линейного программирования два пакета Python:

SciPy – универсальный пакет для научных вычислений с Python. Его внутренний пакет scipy.optimize можно использовать как для линейной, так и для нелинейной оптимизации.

PuLP – API линейного программирования Python для определения задачи и вызова солверов. По умолчанию в качестве солвера используется COIN-OR Branch and Cut Solver (CBC). Еще один отличный солвер с открытым исходным кодом – GNU Linear Programming Kit (GLPK).

Установка SciPy и PuLP

Чтобы следовать этому руководству, вам необходимо установить SciPy и PuLP.

python -m pip install -U scipy pulp

Возможно, вам потребуется запустить pulptest или sudo pulptest, чтобы включить солверы PuLP, особенно если вы используете Linux или Mac:

pulptest

Использование SciPy

В этом разделе мы рассмотрим, как использовать библиотеку SciPy по оптимизации и поиску корней для линейного программирования. Начнём с импорта scipy.optimize.linprog():

from scipy.optimize import linprog

Решение первого примера с помощью SciPy

Начнём с решения примера:

$$\begin{aligned} \text{maximize} \quad & z = x + 2y \\ \text{subject to:} \quad & 2x + y \leq 20 \\ & -4x + 5y \leq 10 \\ & -x + 2y \geq -2 \\ & -x + 5y = 15 \\ & x \geq 0 \\ & y \geq 0 \end{aligned}$$

linprog() решает только задачи минимизации (не максимизации) и не допускает ограничений-неравенств со знаком больше или равно (≥). Чтобы обойти эти проблемы, нам необходимо изменить описание задачи перед запуском оптимизации:

Вместо максимизации z = x + 2y минимизируем отрицательное значение (−z = −x − 2y).

Вместо знака ≥ мы можем умножить «желтое» неравенство на -1 и получить противоположный знак (ограничения по осям рассмотрим далее).

$$\begin{aligned} \text{minimize} \quad & -z = -x - 2y \\ \text{subject to:} \quad & 2x + y \leq 20 \\ & -4x + 5y \leq 10 \\ & x - 2y \leq 2 \\ & -x + 5y = 15 \\ & x \geq 0 \\ & y \geq 0 \end{aligned}$$

На следующем шаге определяем входные значения:

```
obj = [-1, -2]
#
```

```
#      |  └┤ Коэффициент для y
#      └────┤ Коэффициент для x
lhs_ineq = [[ 2,  1],  # левая сторона красного неравенства
            [-4,  5],  # левая сторона синего неравенства
            [ 1, -2]]  # левая сторона желтого неравенства
rhs_ineq = [20,  # правая сторона красного неравенства
            10,  # правая сторона синего неравенства
             2]  # правая сторона желтого неравенства
lhs_eq = [[-1, 5]]  # левая сторона зеленого равенства
rhs_eq = [15]       # правая сторона зеленого равенства
```

Мы поместили значения из системы в соответствующие списки:
obj содержит коэффициенты целевой функции,
lhs_ineq и rhs_ineq содержат коэффициенты из ограничений-неравенств,
lhs_eq и rhs_eq содержат коэффициенты из ограничивающего уравнения.

Примечание

Будьте осторожны с порядком строк и столбцов! Порядок строк для левой и правой сторон ограничений должен быть одинаковым. Каждая строка представляет одно ограничение. Порядок коэффициентов целевой функции и левых частей ограничений должен совпадать. Каждый столбец соответствует одной переменной решения.

Следующим шагом является определение границ каждой переменной. В данном случае они находятся между нулем и положительной бесконечностью:

```
bnd = [(0, float("inf")),  # Границы x
       (0, float("inf"))]  # Границы y
```

Однако эти границы совпадают с установленными по умолчанию в linprog().
Наконец, пришло время оптимизировать и решить интересующую нас проблему:

```
opt = linprog(c=obj, A_ub=lhs_ineq, b_ub=rhs_ineq,
              A_eq=lhs_eq, b_eq=rhs_eq, bounds=bnd,
              method="revised simplex")
opt
```

```
     con: array([0.])
     fun: -16.818181818181817
 message: 'Optimization terminated successfully.'
     nit: 3
   slack: array([ 0.        , 18.18181818,  3.36363636])
  status: 0
 success: True
       x: array([7.72727273, 4.54545455])
```

Параметр c относится к коэффициентам из целевой функции. A_ub и b_ub соответственно связаны с коэффициентами из левой и правой частей ограничений-неравенств. Точно так же A_eq и b_eq относятся к ограничениям уравнений. Параметр bounds служит для указания нижней и верхней границ переменных решения.

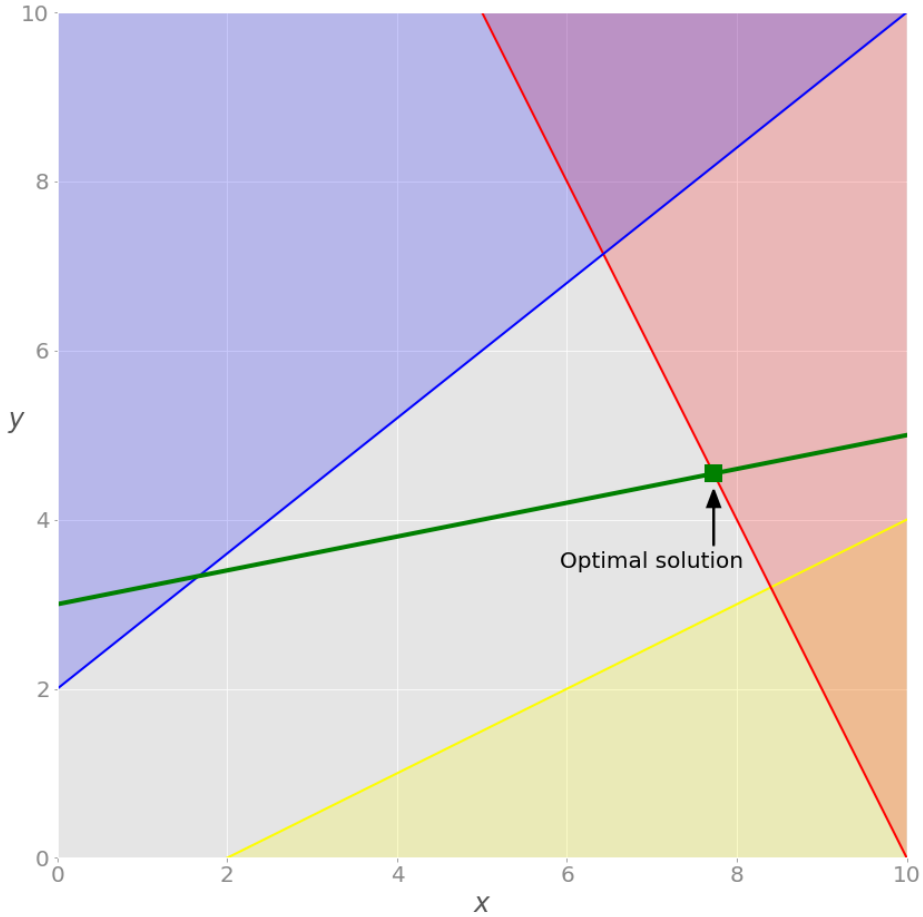
Параметр method определяет используемый алгоритм линейного программирования. Доступны три варианта:

по умолчанию используется метод внутренней точки: method = "inner-point",

измененный двухфазный симплекс-метод method="revised simplex",
симплекс-метод method="simplex"
linprog() возвращает структуру данных со следующими атрибутами:
.con – остатки ограничения-равенства;
.fun – оптимальное значение целевой функции (если найдено);
.message – словесный статус решения;
.nit – количество итераций, необходимых для завершения расчета;
.slack – значения так называемых дополнительных переменных – разниц между значениями левой и правой сторонами ограничений;
.status – целое число от 0 до 4, отражающих результат решения: например, 0, когда было найдено оптимальное решение;
.success – логическое значение, показывающее, найдено ли оптимальное решение;
.x – массив NumPy, содержащий оптимальные значения переменных решения.
Доступ к атрибутам можно получить по отдельности:

```
>>> opt.fun
-16.818181818181817
>>> opt.success
True
>>> opt.x
array([7.72727273, 4.54545455])
```

Графически результат можно отобразить следующим образом.

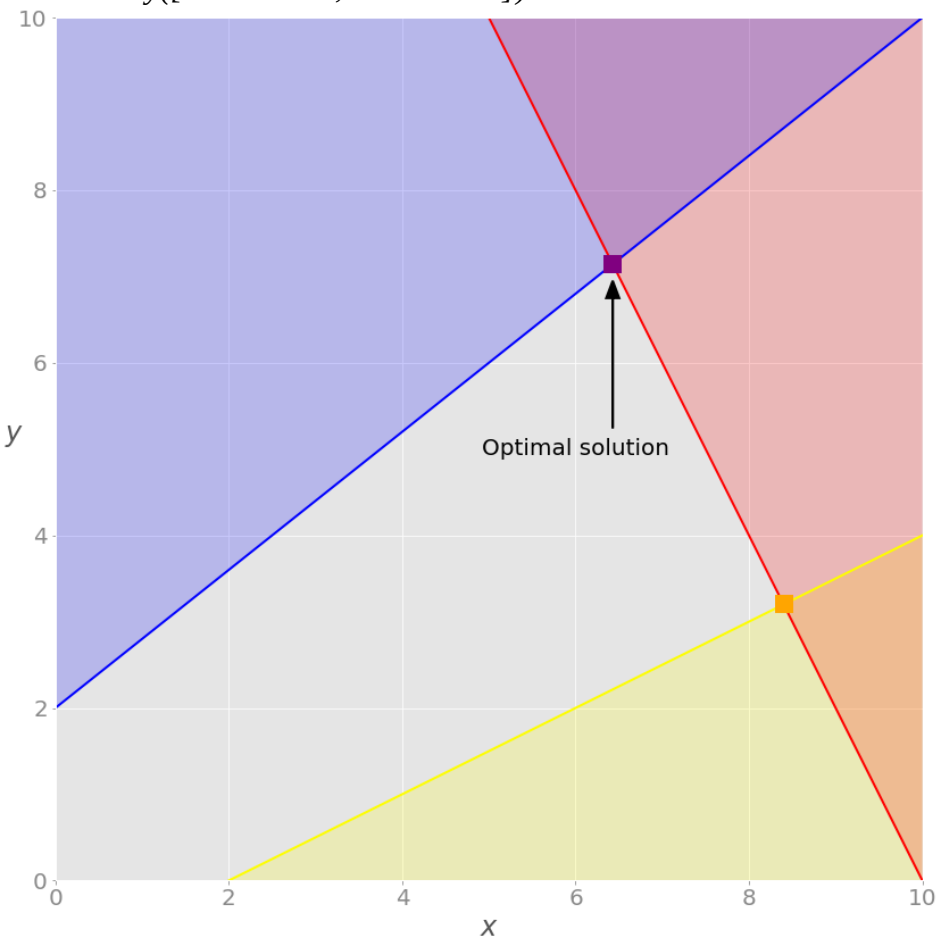


Вначале наша задача органичивалась только неравенствами. Если удалить параметры зеленого уравнения A_eq и b_eq из вызова linprog(), получим следующий результат:

```
opt = linprog(c=obj, A_ub=lhs_ineq, b_ub=rhs_ineq, bounds=bnd,
              method="revised simplex")
```

opt

```
     con: array([], dtype=float64)
     fun: -20.714285714285715
 message: 'Optimization terminated successfully.'
     nit: 2
   slack: array([0.        , 0.        , 9.85714286])
  status: 0
 success: True
       x: array([6.42857143, 7.14285714])
```



Практическая работа 2

АРИФМЕТИЧЕСКИЕ И ЛОГИЧЕСКИЕ ОСНОВЫ РАБОТЫ КОМПЬЮТЕРА

1. **Цель работы**: изучить арифметические и логические основы работы компьютера
2. **Задачи работы:**
   - ознакомиться с арифметическими основами работы компьютера
   - ознакомиться с логическими основами работы компьютера

В соответствии общеобразовательной учебной дисциплиной «Информатика» в результате выполнения заданий ЛР, студент должен:

иметь представление:

- об информационных основах процессов управления;
- о методах поиска информации;
- о принципах кодирования информации;
- о системах счисления;
- о возможности соединения разнотипной информации в одном электронном документе с помощью технологии мультимедиа;

знать:

- различные подходы к определению понятия «информация»;
- методы измерения количества информации: вероятностный и алфавитный. Знать единицы измерения информации;
- назначение наиболее распространенных средств автоматизации информационной деятельности (текстовых редакторов, текстовых процессоров, графических редакторов, электронных таблиц, баз данных, компьютерных сетей);
- назначение и виды информационных моделей, описывающих реальные объекты или процессы;
- использование алгоритма как способа автоматизации деятельности;
- назначение и функции операционных систем.

уметь:

- оценивать достоверность информации, сопоставляя различные источники;
- распознавать информационные процессы в различных системах;
- использовать готовые информационные модели, оценивать их соответствие реальному объекту и целям моделирования;
- осуществлять выбор способа представления информации в соответствии с поставленной задачей;
- иллюстрировать учебные работы с использованием средств информационных технологий;
- создавать информационные объекты сложной структуры;
- просматривать, создавать, редактировать, сохранять записи в базах данных;
- осуществлять поиск информации в базах данных, компьютерных сетях и пр.;
- представлять числовую информацию различными способами (таблица, массив, график, диаграмма и пр.);
- соблюдать правила техники безопасности и гигиенические рекомендации при использовании средств ИКТ.

3. **Подготовка к работе**

Ознакомиться с теоретической справкой.

Подготовить бланк отчета.

4. **Задание**

1) Ознакомиться с теоретической частью
2) Произвести вычисления согласно заданиям

5. **Порядок выполнения работы**

1. Перевести число $7D2,E_{16}$ из шестнадцатеричной системы счисления в двоичную.
2. Перевести число $10111110001,001_2$ из двоичной системы счисления в шестнадцатеричную.

3. Перевести число $11011{,}11_2$ из двоичной системы счисления в десятичную.
4. Перевести число $4A{,}1_{16}$ из шестнадцатеричной системы счисления в десятичную.
5. Перевести число 25_{10} из десятичной системы счисления в двоичную.
6. Перевести число 177_{10} из десятичной системы счисления в шестнадцатеричную
7. Перевести правильную десятичную дробь $0{,}1875_{10}$ в двоичную систему счисления.
8. Перевести правильную десятичную дробь $0{,}47_{10}$ из десятичной системы счисления в шестнадцатеричную с точностью до пяти знаков.
9. Перевести число $9{,}625_{10}$ из десятичной системы счисления в двоичную.
10. Перевести число $399{,}125_{10}$ из десятичной системы счисления в шестнадцатеричную.
11. Перевести число $741{,}625_{10}$ из десятичной системы счисления в шестнадцатеричную.
12. Перевести число DC,216 из шестнадцатеричной системы счисления в десятичную.
13. Выполнить операцию арифметического сложения двоичных чисел $110111{,}01_2$ и $10011{,}1_2$.
14. Перемножить двоичные числа $111{,}1_2$ и 101_2
15. Записать результат выполнения логической операции $\overline{1101101}_2$
16. Логически сложить два двоичных числа 10100010_2 и 1111_2.
17. Логически перемножить два двоичных числа 11110011_2 и 111111_2
18. Доказать равносильность $\overline{(A \wedge \overline{B}) \vee (B \wedge \overline{C})} = (\overline{A} \wedge \overline{B}) \vee (\overline{A} \wedge C) \vee (B \wedge C)$.
19. Упростить высказывание $(A \wedge B \wedge \overline{B}) \vee (A \wedge \overline{A}) \vee (B \wedge C \wedge \overline{C})$

6. Содержание отчёта

20. Название, цель работы
21. Выполнение п.5
22. Выводы по работе.

Приложение

Краткие сведения из теории

Системы счисления

Компьютер предназначен для ввода, обработки, хранения и вывода информации. Любая информация в компьютерах представляется с помощью чисел. Для записи чисел используются системы счисления (десятичная, двоичная, шестнадцатеричная).

Системой счисления называют совокупность приемов и правил наименования и обозначения чисел, с помощью которых можно установить взаимно однозначное соответствие между любым числом и его представлением в виде совокупности конечного числа символов. Она показывает, как мы записываем (представляем) числа и как выполняем над ними действия (сложение, умножение и т.п.). Количество символов, используемых в системе счисления, называют основанием этой системы счисления. Различают непозиционные и позиционные системы.

В непозиционной системе счисления значение каждой цифры в любом месте последовательности цифр, означающей запись числа, не изменяется. Примером непозиционной системы счисления является так называемая римская система счисления. Здесь знак I всегда означает единицу, знак V — пять, знак X — десять. Действительно, в числе XXX, записанном в римской системе счисления, цифра X в любом месте означает десять. Запись чисел в непозиционной системе счисления громоздка и неудобна. Например, число 278 в римской системе счисления запишется в виде CCLXXVIII. В особенности неудобны и сложны в таких системах арифметические действия.

В позиционной системе счисления значение цифры зависит от ее места (позиции) в последовательности цифр, изображающих число. Место для цифры в числе называется разрядом, а количество цифр в числе называется разрядностью числа. Например, десятичное число 2981 является четырехразрядным. Разряды нумеруются справа налево, и каждому разряду соответствует степень основания.

Разряд 3 2 1 0	Название разряда	Степень основания
Число 2 9 8 1		
1	Единицы	10^0
8	Десятки	10^1
9	Сотни	10^2
2	Тысячи	10^3

Примером позиционной системы счисления является десятичная система счисления, которой мы пользуемся. Например, в числе 555 значение цифры 5 зависит от позиции, в которой она находится. Значение цифры 5 в разряде десятков в десять раз больше ее значения в разряде единиц и в десять раз меньше ее значения в разряде сотен.

В вычислительной технике широкое применение нашли также двоичная и шестнадцатеричная системы счисления. В двоичной системе счисления используются два символа: 0 и 1, основание системы равно 2. Двоичная система используется в компьютерах потому, что электрическими сигналами очень просто обозначить двоичные цифры: 0 - нет сигнала и 1 - есть сигнал. В десятичной системе счисления — 10 символов: от 0 до 9. В шестнадцатеричной системе счисления — 16 символов: цифры от 0 до 9 и первые шесть букв латинского алфавита (от A до F). Шестнадцатеричная система счисления просто соотносится с двоичной системой: одна шестнадцатеричная цифра соответствует четырем двоичным разрядам.

Несмотря на то, что десятичная система счисления имеет широкое распространение, электронно-вычислительные машины строятся на двоичных элементах, так как реализовать элементы с десятью четко различимыми состояниями сложно, шестнадцатеричная система счисления используется при составлении программ на языке машинных кодов для более короткой и удобной записи двоичных кодов — команд, данных, адресов и операндов.

Задача перевода из одной системы счисления в другую часто встречается при программировании. Отдельные стандартные процедуры языков программирования Паскаль,

Бейсик, HTML, Си требуют задания параметров в шестнадцатеричной системе счисления. Для непосредственного редактирования данных, записанных на жесткий диск, также необходимо умение работать с шестнадцатеричными числами. Отыскать неисправность в ЭВМ практически невозможно без представления о двоичной системе счисления.

Для обозначения используемой системы счисления нижним индексом указывают ее основание: 1510, 10112, $1EF9016$.

В общем случае в позиционной системе счисления с основанием Q любое число может быть представлено в виде полинома:

$X=A_nQ^n + \ldots + A_1Q^1 + A_0Q^0 + A_{-1}Q^{-1} + A_{-2}Q^{-2} + \ldots + A_{-m}Q^{-m}$, (*)

где в качестве коэффициентов A_i могут стоять любые символы, используемые в данной системе счисления. Принято представлять числа в виде последовательности соответствующих символов:

$X = A_n\ A_{n-1} \ldots A_1\ A_0,\ A_{-1}\ A_{-2} \ldots A_{-m}$

запятая отделяет целую часть числа от дробной.

В таблице приведены двоичные эквиваленты (тетрады) шестнадцатеричных символов:

Двоичная система счисления	Десятичная система счисления	Шестнадцатеричная система счисления
0000	0	0
0001	1	1
0010	2	2
0011	3	3
0100	4	4
0101	5	5
0110	6	6
0111	7	7
1000	8	8
1001	9	9
1010	10	A
1011	11	B
1100	12	C
1101	13	D
1110	14	E
1111	15	F

Рассмотрим правила перевода чисел из одной системы счисления в другую.

Для перевода шестнадцатеричного числа в двоичную систему счисления каждый символ исходного числа заменяется соответствующим четырехразрядным двоичным числом (см. таблицу). В полученном двоичном числе удаляются незначащие нули (крайние слева в целой части и крайние справа — в дробной).

Арифметические основы работы ЭВМ

Двоичная система счисления получила широкое распространение с появлением ЭВМ. Любое число в этой системе представляется сочетанием нулей и единиц. Это позволяет достаточно просто организовать хранение и переработку информации, представленной в двоичном виде. Другим важным достоинством двоичной системы счисления является простота вычислений. Выполнение арифметических действий над числами в двоичной системе счисления производится по тем же правилам, что и в десятичной. При этом пользуются соответствующими таблицами. Рассмотрим только две арифметические операции: сложение и умножение, так как вычитание и деление по существу сводятся к сложению.

+	**0**	**1**		**x**	**0**	**1**
0	0	1		**0**	0	0
1	1	10		**1**	1	1

Правила выполнения арифметических действий над двоичными числами можно свести в таблицу:

Сложение	Умножение
0 + 0 = 0	0 x 0 = 0
0 + 1 = 1	0 x 1 = 0
1 + 0 = 1	1 x 0 = 0
1 + 1 = 10	1 x 1 = 1

В устройствах, реализующих операцию арифметического сложения двоичных чисел, операнды представляют числами определенной разрядности (одинаковой для обоих операндов). При этом неиспользуемые разряды заполняются нулями. Это касается как целой, так и дробной частей числа.

В реальных ЭВМ чаще всего используются 16-, 32- и 64-разрядные числа. Однако для учебных целей при рассмотрении методов выполнения арифметических операций не будем обращать внимание на разрядность операндов (т. е. будем использовать разрядность, отличающуюся от разрядности реальных ЭВМ).

В двоичной системе счисления арифметическое сложение происходит по правилу сложения по модулю два с учетом переноса единицы в старший разряд.

Логические основы работы ЭВМ

Кроме арифметических, ЭВМ выполняют и логические операции, в основе которых положены понятия алгебры логики или, как ее часто называют, булевой алгебры. Основоположником этого раздела математики был Дж. Буль.

Булева алгебра оперирует логическими переменными, которые могут принимать только два значения: истина или ложь, обозначаемые соответственно 1 и 0.

Основной системой счисления ЭВМ является двоичная система счисления, в которой также используются только две цифры: 1 и 0. Таким образом, одни и те же цифровые устройства ЭВМ могут применяться для обработки как числовой информации в двоичной системе счисления, так и логических переменных. Это обуславливает универсальность схемной реализации процесса обработки информации в ЭВМ.

Широкое распространение имеют следующие логические операции: И (логическое умножение), ИЛИ (логическое сложение), НЕ (отрицание). В вычислительной технике они обозначаются соответственно AND (или $\wedge$), OR (или $\vee$), NOT (или $^-$). С помощью этих трех операций можно представить сколь угодно сложную логическую операцию (логическую функцию). Числа, участвующие в логической операции, называются операндами. Операции И, ИЛИ — двухоперандовые. Операция НЕ — однооперандовая.

Рассмотрим пять основных операций алгебры логики.

1. Операция отрицания. Отрицанием утверждения А называется утверждение, которое ложно, если А истинно, и истинно, если А ложно. Отрицание обозначается $\bar{A}$ (читается «не А»). Связь между значением истинности для утверждений А и $\bar{A}$ можно выразить с помощью следующей таблицы истинности для отрицания:

$^-$	A	$\bar{A}$
1	1	0
2	0	1

Из первой строки таблицы видно, что $\bar{A}$ ложно, если А истинно. Вторая строка устанавливает, что $\bar{A}$ истинно, если А ложно.

2. Операция дизъюнкции. Дизъюнкцией утверждений А и В называется утверждение, которое истинно, если истинно хотя бы одно из утверждений А и В, и ложно, когда А и В ложны одновременно. Дизъюнкция обозначается символом А $\vee$ В (читается «А или В») и определяется следующей таблицей истинности:

$\vee$	**A**	**B**	**A $\vee$ B**

1	1	1	1
2	1	0	1
3	0	1	1
4	0	0	0

3. Операция конъюнкции. Конъюнкцией утверждений A и B называется утверждение, которое истинно, если истинны оба утверждения A и B, и ложно – в противном случае, т.е. когда хотя бы одно из утверждений ложно. Конъюнкция обозначается символом $A \wedge B$ (читается «A и B») и определяется следующей таблицей истинности:

∧	**A**	**B**	**A ∧ B**
1	1	1	1
2	1	0	0
3	0	1	0
4	0	0	0

4. Операция эквиваленции. Эквивалентность двух утверждений A и B истинна тогда и только тогда, когда A и B оба истинны или ложны и обозначается $A \sim B$.

~	**A**	**B**	**A ~ B**
1	1	1	1
2	1	0	0
3	0	1	0
4	0	0	1

5. Операция импликации. Импликацией от утверждения A к утверждению B называется утверждение, которое ложно, когда A истинно, а B ложно, и истинно во всех других случаях. Утверждение A называют посылкой, а утверждение B – заключением импликации. Импликация обозначается символом (читается: «из A следует B», « A влечет B», «если A, то B»,) и определяемое следующей таблицей истинности:

⇒	**A**	**B**	$A \Rightarrow B$
1	1	1	1
2	1	0	0
3	0	1	1
4	0	0	1

Истина и ложь могут распределяться между двумя высказываниями четырьмя различными способами.

A	**B**	$A \Rightarrow B$	$\overline{B}$	$\overline{A}$	$\overline{B} \Rightarrow \overline{A}$	$(A \Rightarrow B) \sim (\overline{B} \Rightarrow \overline{A})$
1	1	1	0	0	1	1
1	0	0	1	0	0	1
0	1	1	0	1	1	1
0	0	1	1	1	1	1

Заполнив таблицу истинности, мы получили важный результат: высказывание истинно всегда, т.е. при любом наборе значений истинны и лжи для составляющих его высказываний A и B. Такие высказывания называются тождественно- истинными и обозначаются латинской буквой I. Поэтому можно записать:

Наряду с тождественно-истинными высказываниями существуют высказывания тождественно-ложные, т.е. ложные всегда, независимо от того, истинны или ложны составляющие их высказывания. Тождественно-ложные высказывания обозначают латинской буквой L.

Формулы, имеющие одинаковые таблицы истинности, назовем эквивалентными. Эквивалентные формулы алгебры высказываний – аналог тождественных выражений обычной алгебры. Так как таблицы истинности конечны, то эквивалентность формул в алгебре высказываний можно доказать с помощью их таблиц истинности, сравнив их. Этот метод практически приемлем только в случае небольшого числа простых высказываний, образующих составные. Ведь если сложное высказывание состоит из n простых, то таблица истинности такого высказывания содержит 2n строк, что при n = 10, например, превзойдет тысячу. В то же время в приложениях алгебры логики, в частности в теории автоматического управления при анализе релейно-контактных и электронно-ламповых схем, как раз приходится иметь дело с высказываниями, составленными из сотен и даже тысяч простых высказываний. Доказательство равносильности высказываний с помощью таблиц истинности в таких случаях практически невозможно.

Равносильность высказываний можно устанавливать и другим способом: некоторое количество основных равносильностей проверяется на основании таблиц истинности, полученные равенства используются при доказательстве других равенств с помощью основных тождеств алгебры высказываний.

Наиболее важными «тождествами» алгебры высказываний являются следующие:

1. Закон двойного отрицания

$$\overline{\overline{A}} = A$$

2. Коммутативность дизъюнкции

$$A \vee B = B \vee A$$

3. Коммутативность конъюнкции

$$A \wedge B = B \wedge A$$

4. Ассоциативность дизъюнкции

$$A \vee (B \vee C) = (A \vee B) \vee C$$

5. Ассоциативность конъюнкции

$$A \wedge (B \wedge C) = (A \wedge B) \wedge C$$

6. Первый дистрибутивный закон

$$A \wedge (B \vee C) = (A \wedge B) \vee (A \wedge C)$$

7. Второй дистрибутивный закон

$$A \vee (B \wedge C) = (A \vee B) \wedge (A \vee C)$$

8. Законы де Моргана

$$\overline{A \vee B} = \overline{A} \wedge \overline{B}$$

9. Законы идемпотентности

$$A \vee A = A, \quad A \wedge A = A.$$

10. Законы, включающие тождественно-истинные (I) и тождественно-ложные (L) высказывания

$$A \vee \overline{A} = I, \quad A \wedge \overline{A} = L,$$

$$A \vee I = I, \quad A \wedge I = A, \quad \overline{I} = L.$$

$$A \vee L = A, \quad A \wedge L = L,$$

Введенные пять основных логических операций не являются независимыми: одни из них могут быть выражены через другие. В частности, эквиваленция и импликация выражаются через дизъюнкцию, конъюнкцию и отрицание следующим образом:

$$A \Rightarrow B = \overline{A} \vee B,$$

$$A \sim B = (A \wedge B) \vee (\overline{A} \wedge \overline{B}).$$

Практическая работа 3
ПОНЯТИЕ АЛГОРИТМА. СВОЙСТВА АЛГОРИТМА. СПОСОБЫ ЗАПИСИ АЛГОРИТМОВ

1. Цель работы: изучить понятие алгоритма, ознакомиться с его свойствами и способами записи.

2. Задачи работы:

- уметь создавать информационные объекты сложной структуры
- знать назначение наиболее распространенных средств автоматизации информационной деятельности
- научиться создавать презентации средствами PowerPoint, форматировать редактировать слайды.

В соответствии общеобразовательной учебной дисциплиной «Информатика» в результате выполнения заданий ПР, студент должен:

иметь представление:

- о методах поиска информации;
- о принципах кодирования информации;
- о возможности соединения разнотипной информации в одном электронном документе с помощью технологии мультимедиа;

знать:

- различные подходы к определению понятия «информация»;
- назначение наиболее распространенных средств автоматизации информационной деятельности (текстовых редакторов, текстовых процессоров, графических редакторов, электронных таблиц, баз данных, компьютерных сетей);
- назначение и виды информационных моделей, описывающих реальные объекты или процессы;
- использование алгоритма как способа автоматизации деятельности;

уметь:

- оценивать достоверность информации, сопоставляя различные источники;
- использовать готовые информационные модели, оценивать их соответствие реальному объекту и целям моделирования;
- осуществлять выбор способа представления информации в соответствии с поставленной задачей;
- иллюстрировать учебные работы с использованием средств информационных технологий;
- создавать информационные объекты сложной структуры;
- представлять числовую информацию различными способами (таблица, массив, график, диаграмма и пр.);
- соблюдать правила техники безопасности и гигиенические рекомендации при использовании средств ИКТ.

3. Подготовка к работе

Для обеспечения выполнения работы необходимо иметь компьютер со следующим обеспечением: операционная система Windows и MS Office 2007 и выше.

4. Задание

1) Ознакомиться с теоретической частью
2) Изучить алгоритм как понятие
3) Научиться записывать алгоритм разными способами

5. Порядок выполнения работы

Записать в виде блок схемы следующие задания:

1. Задавая a и b, вычислить значения функции z = sin x + cos y, где x = a + cos|a 2− 2ab|, y = a − cos|a 2− 2ab|.
2. Составить алгоритм вычисления функции

$$z=\begin{cases}\ln x, & \text{если } x\ge\pi,\\ \sqrt{|3+x|}, & \text{если } -2\pi<x<\pi,\\ \sin x, & \text{если } x\le -2\pi.\end{cases}$$

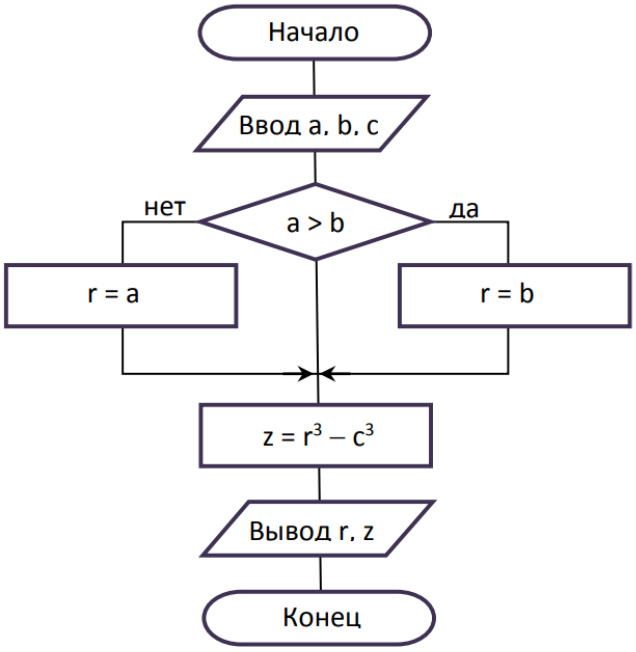
3. Составить алгоритм вычисления суммы ряда:

$$S=\sin x+\frac{\sin 3x}{3}+\frac{\sin 5x}{5}+\ldots=\sum_{n=1}^{\infty}\frac{\sin(2n-1)x}{2n-1}$$

с точностью до члена ряда, меньшего ε, для заданного значения x

4. У студента имеются накопления в сумме S рублей. Ежемесячная стипендия составляет A рублей, расходы на проживание превышают стипендию и составляют B рублей в месяц накануне начала учебы. Рост цен ежемесячно увеличивает расходы на 2% по сравнению с расходами предыдущего месяца. Определить количество месяцев, которые может прожить студент, используя только накопления и стипендию.
5. Шаг 1. Ввести значения t, z.
   Шаг 2. Проверить условие t □ z. Если условие выполняется, то перейти к шагу 4.
   Если условие не выполняется, то перейти к шагу 3.
   Шаг 3. Вычислить значение $p = t^2 + z^2$ и перейти к шагу 5.
   Шаг 4. Вычислить значение $p = t^2 - z^2$.
   Шаг 5. Напечатать результат p.
   Шаг 6. Прекратить вычисления.

Опишите словесно алгоритм, приведенный на рисунке:



Записать в виде алгоритма:
Предприниматель, начав дело, взял кредит размером S рублей под p процентов годовых и вложил в свое дело. По прогнозам, его дело должно давать прибыль R рублей в год. Сможет ли он накопить сумму, достаточную для погашения кредита, и если да, то через сколько лет?

6. Содержание отчёта

6. Название, цель работы
7. Выполнение п.5
8. Выводы по работе.

7. Контрольные вопросы к защите

1. Дайте определение алгоритма.
2. Можно ли считать алгоритмом: (a) правила правописания; (b) законы физики; (c) математические формулы; (d) статьи уголовного кодекса. Ответы обоснуйте.
3. В повседневной жизни существует множество синонимов понятия «алгоритм». Что из перечисленного ниже нельзя назвать алгоритмом? Ответ обоснуйте.

1) Рецепт приготовления блюда.
2) Инструкцию по использованию бытового прибора.
3) Афишу кинотеатра.
4) План создания презентации, предлагаемый Мастером автосодержания.

4. Какими способами можно описать алгоритм решения задачи?
5. Какую форму записи алгоритма называют блок-схемой? Приведите пример.
6. В чем суть таких свойств алгоритма как «результативность» и «массовость»? Обладает ли требованиям массовости и результативности следующая последовательность действий при вычислении значения функции y = (a +b)/c:

Шаг 1. Ввести значения переменных a, b и c.
Шаг 2. Вычислить значение функции y = (a +b)/c.
Шаг 3. Напечатать значение результата y.
Шаг 4. Прекратить вычисления.

7. Какой алгоритм называют линейным? Приведите пример.
8. Какой алгоритм называется циклическим?

Приложение

Краткие сведения из теории

Алгоритм — это определённая последовательность действий, которые необходимо выполнить, чтобы получить результат. Алгоритм может представлять собой некоторую последовательность вычислений, а может - последовательность действий нематематического характера. Для любого алгоритма справедливы общие закономерности - свойства алгоритма.

- Свойства алгоритма.
- Дискретность.
- Понятность
- Детерминированность
- Массовость
- Результативность

Дискретность - это свойство алгоритма, когда алгоритм разбивается на конечное число элементарных действий (шагов).

Понятность - свойство алгоритма, при котором каждое из этих элементарных действий (шагов) являются законченными и понятными.

Детерминированность - свойство, когда каждое действие (операция.указание.шаг.требование) должно пониматься в строго определённом смысле, чтобы не оставалась места произвольному толкованию. чтобы каждый, прочитавший указание, понимал его однозначно.

Массовость - свойство, когда по данному алгоритму должна решаться не одна, а целый класс подобных задач.

Результативность – свойство, при котором любой алгоритм в процессе выполнения должен приводить к определённому результату. Отрицательный результат также является результатом.

Существуют различные способы записи алгоритмов:

- Словесно-формульная,
- Блок-схема,
- Псевдокод (алгоритмические языки)

Алгоритм может быть записан различными способами: на естественном языке в виде описания; в виде графических блок-схем; на специальном алгоритмическом языке. Запись алгоритмов на родном языке доступна и удобна. Примеров таких записей множество, хотя бы книга кулинарных рецептов есть не что иное, как сборник алгоритмов, написанных на родном языке. Существенным недостатком такой записи является недостаточная наглядность, что особенно сказывается, когда алгоритм имеет много ветвлений.

Словесно-формульная запись алгоритма

Для исполнителя-человека алгоритм может быть представлен предложениями обычного текста с использованием в случае необходимости математической или другой символики. Такой способ записи алгоритма называют словесно-формульным.

Запись в виде блок-схемы.

Блок-схема алгоритма – это его графическое изображение в виде схемы связанных между собой с помощью линий перехода блоков – специальных графических объектов, каждый из которых соответствует определенным шагам алгоритма.

Внутри блока дается описание соответствующих действий. Блоки, как правило, располагаются сверху вниз. Линии соединения отдельных блоков показывают направление процесса обработки в схеме. Стрелки на соединяющих линиях не ставятся при направлениях сверху вниз и слева направо; противоположные направления показывают стрелкой на линии.

При изображении блок-схем используются следующие типы блоков:

Все имеющиеся алгоритмы можно разделить на три вида:

- линейные алгоритмы;
- алгоритмы ветвления;
- циклические алгоритмы.

Линейная последовательность (следование) – это набор действий, выполняемых последовательно друг за другом. На блок-схеме ему соответствует несколько расположенных друг за другом блоков обработки и/или блоков ввода-вывода.

Выбор (разветвление) – это набор действий, начинающийся с проверки некоторого условия. Результатом проверки может быть один из двух исходов: условие выполнено («ДА») или условие не выполнено («НЕТ»). Каждому из этих исходов могут соответствовать некоторые действия или отсутствие каких-либо действий для одного из исходов. Обычно условие формулируется в виде логического выражения, вычисление которого в зависимости от результатов предшествующих действий дает значение истина («ДА») или ложь («НЕТ»).

Повторение (цикл) – это набор действий, которые могут выполняться многократно. Этот набор включает проверку условия, определяющего количество повторений цикла. Различают три варианта циклов:

- цикл с предусловием,
- цикл с постусловием,
- цикл с параметром.

Цикл с предусловием («цикл-пока») проверка условия является первым выполняемым действием; если результат проверки – «ДА», то выполняются другие действия цикла (тело цикла) и затем вновь проверка условия и т. д.; если результат проверки условия – «НЕТ», то цикл заканчивается. Действия цикла, следующие за проверкой условия, могут ни разу не выполниться, если результат первой проверки условия – «НЕТ».

В цикле с постусловием («Цикл-до») проверка условия является последним выполняемым в составе цикла действием; если результат проверки – «ДА», то цикл заканчивается; если результат проверки условия – «НЕТ», то вновь выполняются все предшествующие действия цикла и т. д. В отличие от цикла с предусловием все действия в цикле с постусловием всегда, по крайне мере, один раз выполняются.

Для того чтобы циклы с пред- или постусловием заканчивались после какого-то количества повторений, действия внутри циклов должны изменять некоторые величины, влияющие на результат проверки условия. Само количество повторений таких циклов может быть заранее, до начала их выполнения, неизвестно. В тех случаях, когда число повторений известно заранее или определяется до начала цикла, может быть использован цикл с параметром.

Такой цикл начинается с заголовка, содержащего параметр цикла с указанием его начального и конечного значений; за заголовком следуют действия, повторяемые в этом цикле. При каждом повторении цикла параметр возрастает (или убывает) на некоторую постоянную величину (шаг), изменяясь от начального до конечного значения.

Практическая работа 4

ПРОГРАММИРОВАНИЕ РАЗВЕТВЛЯЮЩИХСЯ АЛГОРИТМОВ

1. Цель работы: сформировать навыки и умения при составлении программ с операторами условного перехода в Python. Изучить правила программирования алгоритмов со структурой ветвления.

2. Задачи работы:

- познакомиться со структурой ветвления в Python
- познакомиться со структурами if, elif, else

В соответствии общеобразовательной учебной дисциплиной «Информатика» в результате выполнения заданий ЛР, студент должен:

иметь представление:

- о принципах кодирования информации; о системах счисления;

знать:

- различные подходы к определению понятия «информация»;
- методы измерения количества информации: вероятностный и алфавитный. Знать единицы измерения информации;

уметь:

- оценивать достоверность информации, сопоставляя различные источники;
- использовать готовые информационные модели, оценивать их соответствие реальному объекту и целям моделирования;
- осуществлять выбор способа представления информации в соответствии с поставленной задачей;
- представлять числовую информацию различными способами (таблица, массив, график, диаграмма и пр.).

3. Подготовка к работе

Ознакомиться с теоретической справкой

4. Задание

1) Ознакомиться с теоретической справкой
2) Выполнить задание из приложения 2, в соответствии с вариантом
3) Подготовиться к ответу на контрольные вопросы

5. Порядок выполнения работы

1. Ознакомиться с теоретической справкой
2. Выполнить задание 1
3. Представить преподавателю отчет

6. Содержание отчёта

1. Название, цель работы
2. Выполнение п.5 ()
3. Выводы по работе.

7. Контрольные вопросы

1. Дайте определение разветвляющегося алгоритма.
2. Напишите сокращенную форму условного перехода.
3. Переведите на язык Python следующие команды:

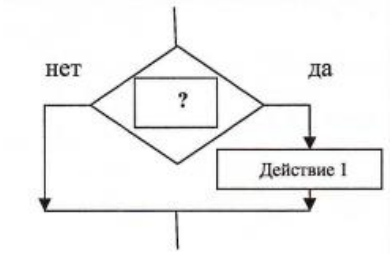
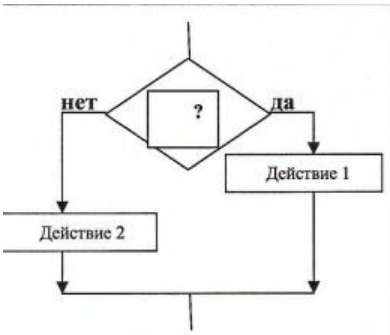
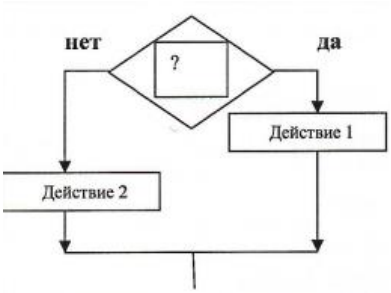
а) если a=5, то выводим на экран «отлично», если a=4, то выводим на экран «хорошо», иначе выводим на экран « вы не прошли в финал»;

б) если x<=1, то y=1+x+x^2 , иначе y= (1 + x)^(½).

Приложение

Теоретическая справка

Разветвляющийся алгоритм - алгоритм, в котором проверяется условие, в зависимости от которого выполняется то или иное действие.

Элементы блок-схемы	**Назначение**	**Соответствующий оператор**
	Сокращенный условный оператор. Внутри блока записывается условие. Если условие выполняется, то одно действие	if: <условие> <действие> else:
	Полный условный оператор. Если условие выполняется, то выполняется одно Действие 1>, иначе Действие 2	if: <условие> <действие> else: <действие>
	Если условие выполняется, то выполняется несколько действий в ветви «ДА», иначе несколько действий в ветви «НЕТ»	if: <условие> <действие> elif: <условие> <действие> else: <действие>

Приложение 2

Задание 1

Составить блок- схему и программу вычисления следующего выражения:

$$Y=\begin{cases} x^2-3x+9 & \text{если } x\le 3 \\ \dfrac{1}{x^3+6} & \text{если } x<3 \end{cases}$$ Вариант №1

$$Y=\begin{cases} -x^2+3x+9 & \text{если } x\ge 3 \\ \dfrac{1}{x^3-6} & \text{если } x>3 \end{cases}$$ Вариант №2

$$Y=\begin{cases} 9 & \text{если } x\le -3 \\ \dfrac{1}{x^2+1} & \text{если } x<-3 \end{cases}$$ Вариант №3

$$Y=\begin{cases} 0 & \text{если } x\le 1 \\ \dfrac{1}{x+6} & \text{если } x>1 \end{cases}$$ Вариант №4

$$Y=\begin{cases} -3x+9 & \text{если } x\le 7 \\ \dfrac{1}{x-7} & \text{если } x>7 \end{cases}$$ Вариант №5

$$Y=\begin{cases} 3x-9 & \text{если } x\le 7 \\ \dfrac{1}{x^2-4} & \text{если } x>7 \end{cases}$$ Вариант №6

$$Y=\begin{cases} x^2+4x+5 & \text{если } x\le 2 \\ \dfrac{1}{x^2+4x+5} & \text{если } x>2 \end{cases}$$ Вариант №7

$$Y=\begin{cases} -x^2+x-9 & \text{если } x\ge 8 \\ \dfrac{1}{x^4-6} & \text{если } x<8 \end{cases}$$ Вариант №8

Вариант №9

$$Y=\begin{cases}4x^2+2x-19 & \text{если } x\geq -3{,}5\\ -\dfrac{2x}{-4x+1} & \text{если } x<3{,}5\end{cases}$$

Вариант №10

$$Y=\begin{cases}-x^2+3x+9 & \text{если } x\geq 3\\ \dfrac{x}{x^2+1} & \text{если } x<3\end{cases}$$

Вариант №11

$$Y=\begin{cases}1{,}2x^2-3x-9 & \text{если } x>3\\ \dfrac{12{,}1}{2x^2+1} & \text{если } x\leq 3\end{cases}$$

Вариант №12

$$Y=\begin{cases}x^2+3x+9 & \text{если } x\leq 3\\ \dfrac{\sin x}{x^2-9} & \text{если } x>3\end{cases}$$

Вариант №13

$$Y=\begin{cases}\cos 2x+9 & \text{если } x>-4\\ -\dfrac{\cos x}{x-9} & \text{если } x\leq -4\end{cases}$$

Практическая работа 5

Простейшие программы на языке программирования Python. Арифметические выражения на Python. Ввод, вывод

1. **Цель работы:** познакомиться со средой разработки Python. Изучить основные типы данных, команды ввода и вывода данных, познакомиться с основными математическими операциями в Python
2. **Задачи работы:**

- знать типы алгоритмов;
- научиться создавать алгоритмы.

В соответствии общеобразовательной учебной дисциплиной «Информатика» в результате выполнения заданий ЛР, студент должен:

иметь представление:

- о принципах кодирования информации; о системах счисления;

знать:

- различные подходы к определению понятия «информация»;
- методы измерения количества информации: вероятностный и алфавитный. Знать единицы измерения информации;

уметь:

- оценивать достоверность информации, сопоставляя различные источники;
- использовать готовые информационные модели, оценивать их соответствие реальному объекту и целям моделирования;
- осуществлять выбор способа представления информации в соответствии с поставленной задачей;
- представлять числовую информацию различными способами (таблица, массив, график, диаграмма и пр.).

3. **Подготовка к работе**

Для подготовки к работе, ознакомьтесь с теоретическим материалом из кратких сведений из теории

4. **Задание**

1) Напишите программу, которая запрашивала бы у пользователя: Имя, Фамилия, Возраст

- фамилия, имя ("Ваши фамилия, имя?")
- возраст ("Сколько Вам лет?")

После этого выводила бы три строки:

- "Ваши фамилия, имя"
- «Ващ возраст»
- "Ваш возраст"

2) Повторите вычисления из примера часть 2

5. **Порядок выполнения работы**

- Ознакомьтесь теоретической справкой
- Выполните задание
- Результаты работы представьте в формате очета

6. **Содержание отчета**

a. Название, цель работы, вариант
b. Листинг программы
c. Выводы по работе

Краткие сведения из теории

Python– это объектно-ориентированный, интерпретируемый, переносимый язык сверхвысокого уровня. Программирование на Python позволяет получать быстро и качественно необходимые программные модули.

В комплекте вместе с интерпретатором Python идет IDLE (интегрированная среда разработки). По своей сути она подобна интерпретатору, запущенному в интерактивном режиме с расширенным набором возможностей (подсветка синтаксиса, просмотр объектов, отладка и т.п.).

Для запуска IDLE в Windows необходимо перейти в папку Python в меню “Пуск” и найти там ярлык с именем “IDLE (Python 3.X XX-bit)”.

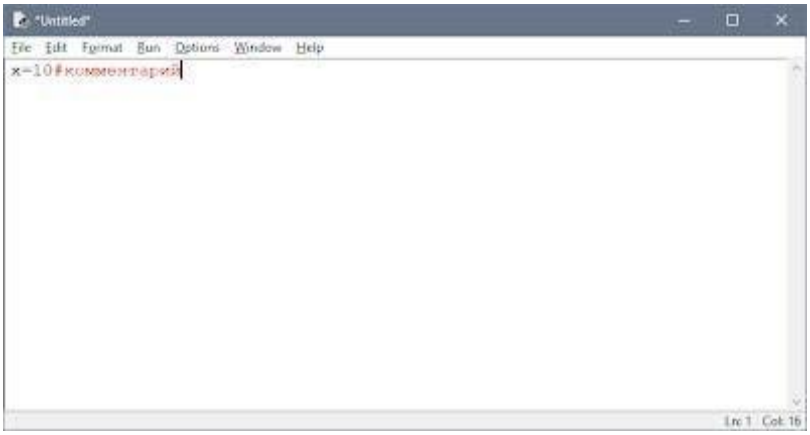
Для запуска редактора программы (кода) следует выполнить команду File->New File или сочетание клавиш Ctrl+N.

Любая Python-программа состоит из последовательности допустимых символов, записанных в определенном порядке и по определенным правилам.

Программа включает в себя:

- комментарии;
- команды;
- знаки пунктуации;
- идентификаторы;
- ключевые слова.

Комментарии в Python обозначаются предваряющим их символом # и продолжаются до конца строки (т.е. в Python все комментарии являются однострочными), при этом не допускается использование перед символом # кавычек:



Знаки пунктуации

В алфавит Python входит достаточное количество знаков пунктуации, которые используются для различных целей. Например, знаки "+" или " *" могут использоваться для сложения и умножения, а знак запятой "," - для разделения параметров функций.

Идентификаторы

Идентификаторы в Python это имена используемые для обозначения переменной, функции, класса, модуля или другого объекта.

Ключевые слова

Некоторые слова имеют в Python специальное назначение и представляют собой управляющие конструкции языка.

Ключевые слова в Python:

['False', 'None', 'True', 'and', 'as', 'assert', 'break', 'class', 'continue', 'def', 'del', 'elif', 'else', 'except', 'finally', 'for', 'from', 'global', 'if', 'import', 'in', 'is', 'lambda', 'nonlocal', 'not', 'or', 'pass', 'raise', 'return', 'try', 'while', 'with', 'yield']

Типы данных

1. None (неопределенное значение переменной)
2. Логические переменные (Boolean Type)
3. Числа (Numeric Type)
    1. int – целое число
    2. float – число с плавающей точкой
    3. complex – комплексное число
4. Списки (Sequence Type)
    1. list – список
    2. tuple – кортеж
    3. range – диапазон
    4. Строки (Text Sequence *Type*)
    5. str

Ввод и вывод данных

Ввод данных осуществляется при помощи команды **input(**список ввода**)**: a = input()
print(a)

В скобках функции можно указать сообщение - комментарий к вводимым данным:

```
a = input ("Введите количество: ")
```

Команда input() по умолчанию воспринимает входные данные как строку символов. Поэтому, чтобы ввести целочисленное значение, следует указать тип данных int():

```
a = int (input())
```

Для ввода вещественных чисел применяется команда

```
a=float(input())
```

Вывод данных осуществляется при помощи команды **print**(список вывода):

```
a = 1
b = 2
print(a)
print(a + b)
print('сумма = ', a + b)
```

Существует возможность записи команд в одну строку, разделяя их через **;**. Однако не следует часто использовать такой способ, это снижает удобочитаемость:

```
a = 1; b = 2;
print(a) print (a + b)
print ('сумма = ', a + b)
```

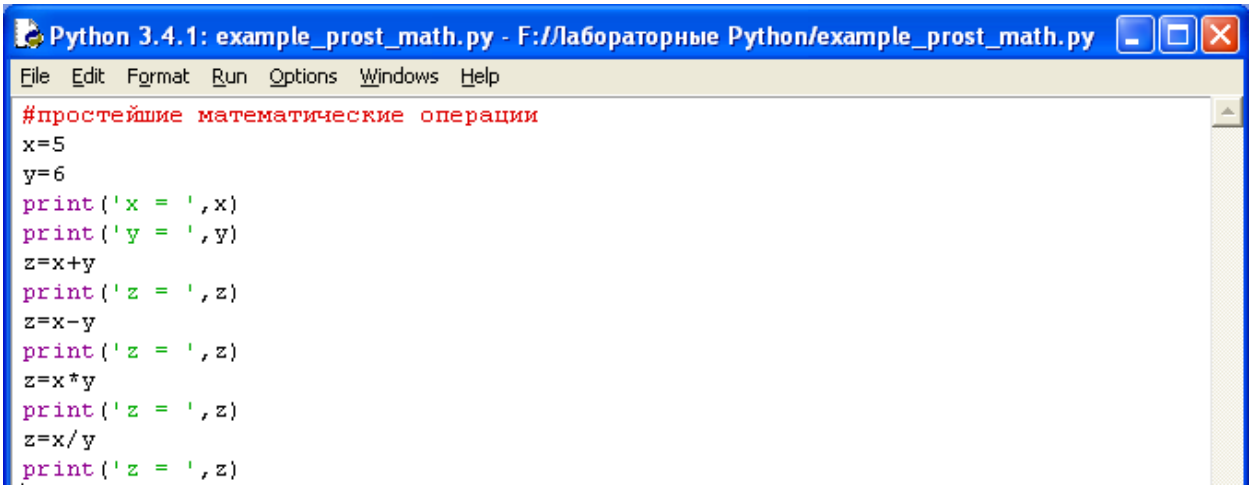
Для команды **print** может задаваться так называемый сепаратор — разделитель между элементами вывода:

```
x
=2
y=5
print ( x, "+", y, "=", x+y, sep = " " )
```

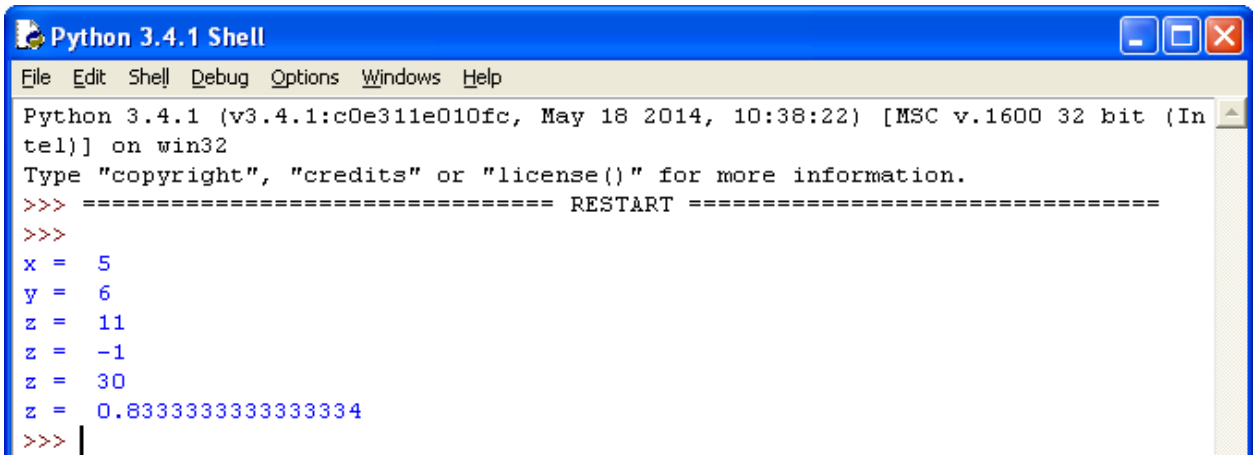
Результат отобразится с пробелами между элементами: 2 + 5 = 7

Простые арифметические операции над числами

x + y	Сложение
x - y	Вычитание
x * y	Умножение
x / y	Деление



Пример программы на Python



Результат выполнения программы с применением простых арифметических операций

Для форматированного вывода используется **format**:

Строковый метод format() возвращает отформатированную версию строки, заменяя идентификаторы в фигурных скобках {}. Идентификаторы могут быть позиционными, числовыми индексами, ключами словарей, именами переменных.

Синтаксис команды **format**:

поле замены := "{" [имя поля] ["!" преобразование] [":" спецификация] "}"

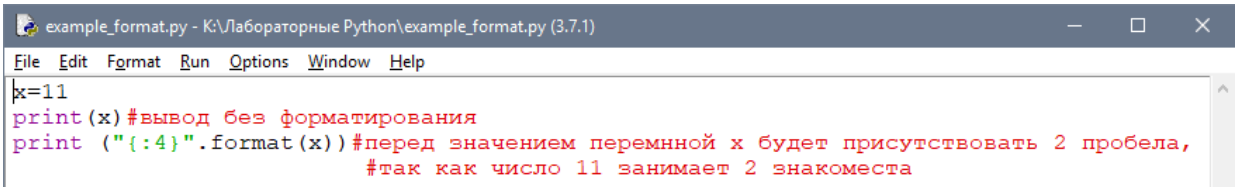
имя поля := arg_name ("." имя атрибута | "[" индекс "]")*

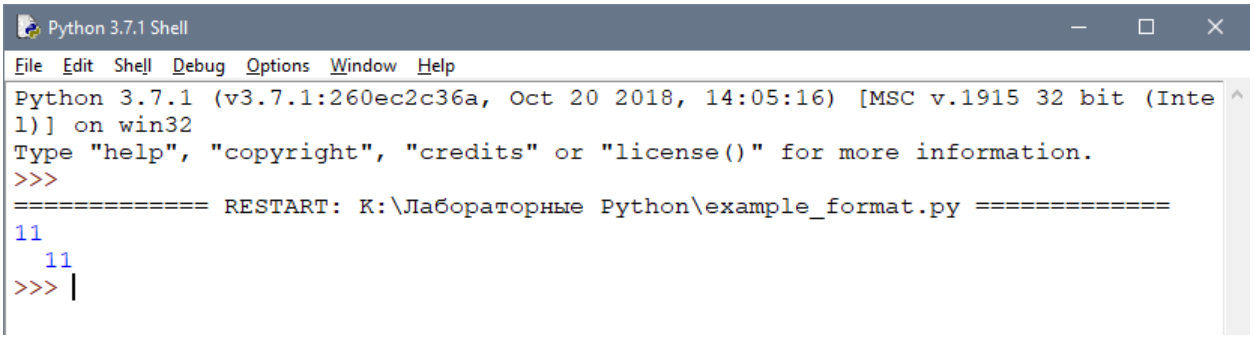
преобразование := "r" (внутреннее представление) | "s" (человеческое представление)

спецификация := см. ниже

Аргументов в format() может быть больше, чем идентификаторов в строке. В таком случае оставшиеся игнорируются.

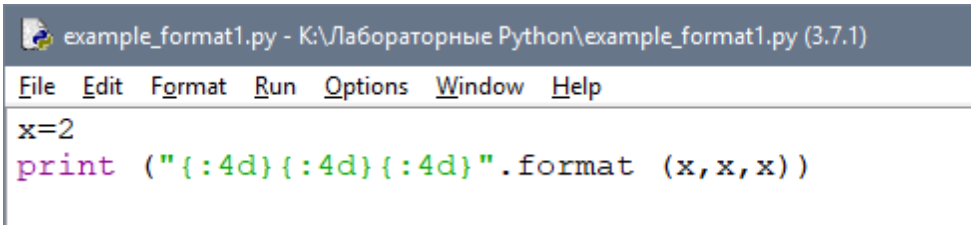
Идентификаторы могут быть либо индексами аргументов, либо ключами:

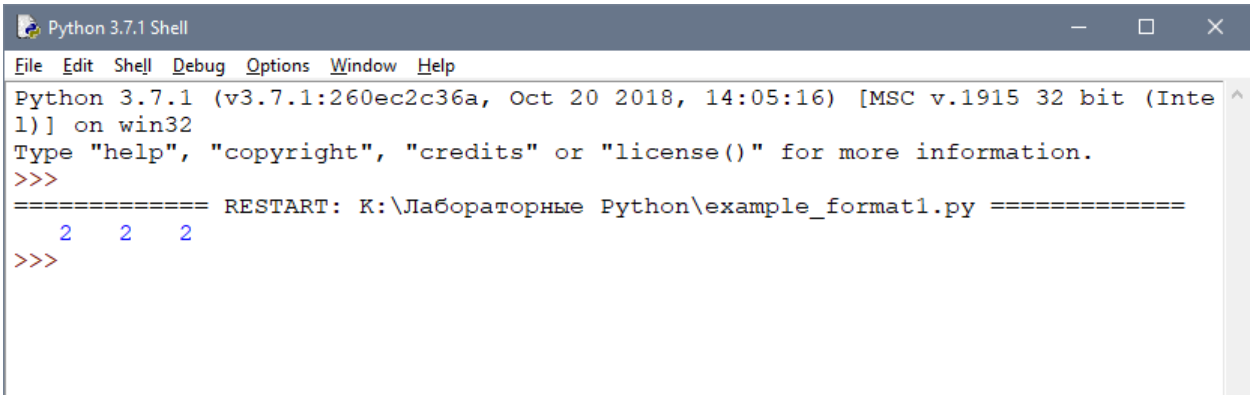




В результате выведется число 11, а перед ним два пробела, так как указано использовать для вывода четыре знакоместа.

Или с несколькими аргументами:





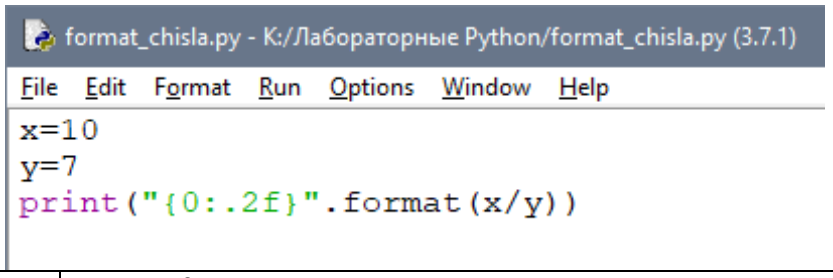
В итоге каждое из значений выводится из расчета 4 знакоместа.
Спецификация формата:

точность	:= integer
тип	:= "b" | "c" | "d" | "e" | "E" | "f" | "F" | "g" | "G" | "n" | "o" | "s" | "x" | "X" | "%"

Тип	**Значение**
'd', 'i', 'u'	Десятичное число.
'o'	Число в восьмеричной системе счисления.
'x'	Число в шестнадцатеричной системе счисления (буквы в нижнем регистре).
'X'	Число в шестнадцатеричной системе счисления (буквы в верхнем регистре).
'e'	Число с плавающей точкой с экспонентой (экспонента в нижнем регистре).
'E'	Число с плавающей точкой с экспонентой (экспонента в верхнем регистре).
'f', 'F'	Число с плавающей точкой (обычный формат).
'g'	Число с плавающей точкой. с экспонентой (экспонента в нижнем регистре), если она меньше, чем -4 или точности, иначе обычный формат.
'G'	Число с плавающей точкой. с экспонентой (экспонента в верхнем регистре), если она меньше, чем -4 или точности, иначе обычный формат.
'c'	Символ (строка из одного символа или число - код символа).
's'	Строка.
'%'	Число умножается на 100, отображается число с плавающей точкой, а за ним знак %.

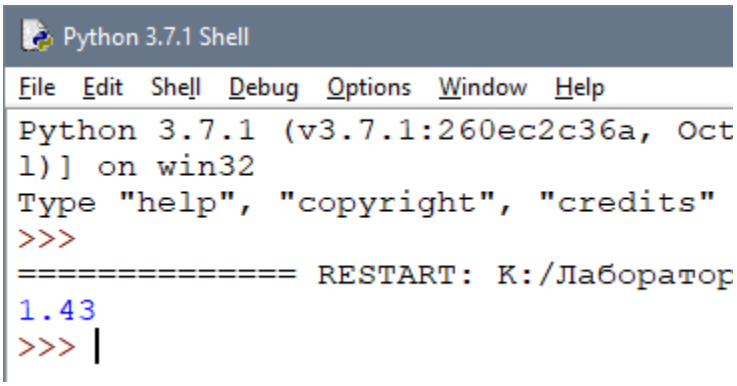
Для форматирования вещественных чисел с плавающей точкой используется следующая команда:

```
print('{0:.2f}'.format(вещественное число))
```



спецификация	:= [[fill]align][sign][#][0][width][,][.precision][type]
заполнитель	:= символ кроме '{' или '}'
выравнивание	:= "<" | ">" | "=" | "^"
знак	:= "+" | "-" | " "
ширина	:= integer

В результате выведется число с двумя знаками после запятой.



Пример

Напишите программу, которая запрашивала бы у пользователя:

- ФИО ("Ваши фамилия, имя, отчество?")

- возраст ("Сколько Вам лет?")

- место жительства ("Где вы живете?")

После этого выводила бы три строки:
"Ваше имя"
"Ваш возраст" "Вы живете в"

Решение

```
a=input('Введите ваши фамилию, имя, отчество ')
b=input('Сколько вам лет? ')
c=input('Где вы живёте? ')
print('Ваше имя ',a)
print('Ваш возраст ',b)
print('Вы живете в ',c)
```

```
Введите ваши фамилию, имя, отчество Иванов Иван Иванович
Сколько вам лет? 15
Где вы живёте? Уссурийск
Ваше имя  Иванов Иван Иванович
Ваш возраст  15
Вы живете в  Уссурийск
```

Целые числа (int)

Числа в Python 3 поддерживают набор самых обычных математических операций:

x + y	Сложение
x - y	Вычитание
x * y	Умножение
x / y	Деление
x // y	Получение целой части от деления
x % y	Остаток от деления
-x	Смена знака числа
abs(x)	Модуль числа
divmod(x, y)	Пара (x // y, x % y)
x ** y	Возведение в степень
pow(x, y[, z])	x : Число, которое требуется возвести в степень. y : Число, являющееся степенью, в которую нужно возвести первый аргумент. Если число отрицательное или одно из чисел "x" или "y" не целые, то аргумент "z" не принимается. z : Число, на которое требуется произвести деление по модулю. Если число указано, ожидается, что "x" и "y" положительны и имеют тип int.

Пример применения вышеописанных операций над целыми числами

```
x = 5
y = 2
z = 3
x+y = 7
x-y = 3
x*y = 10
x/y = 2.5
x//y = 2
x%y = 1
-x= -5
abs(-x) = 5
divmod(x,y) = (2, 1)
x**y = 25
pow(x,y,z) = 1
```

Часть 2

Вещественные числа (float)

Вещественные числа поддерживают те же операции, что и целые. Однако (из- за представления чисел в компьютере) вещественные числа неточны, и это может привести к ошибкам.

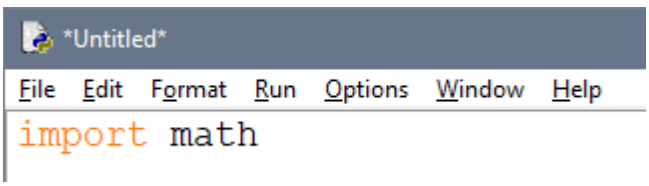
Пример применения вышеописанных операций над вещественными числами

```
x = 5.5
y = 2.3
x+y = 7.8
x-y = 3.2
x*y = 12.649999999999999
x/y = 2.3913043478260874
x//y = 2.0
x%y = 0.9000000000000004
-x= -5.5
abs(-x) = 5.5
divmod(x,y) = (2.0, 0.9000000000000004)
x**y = 50.44686540422945
```

Библиотека (модуль) math

В стандартную поставку Python входит библиотека math, в которой содержится большое количество часто используемых математических функций.

Для работы с данным модулем его предварительно нужно импортировать.

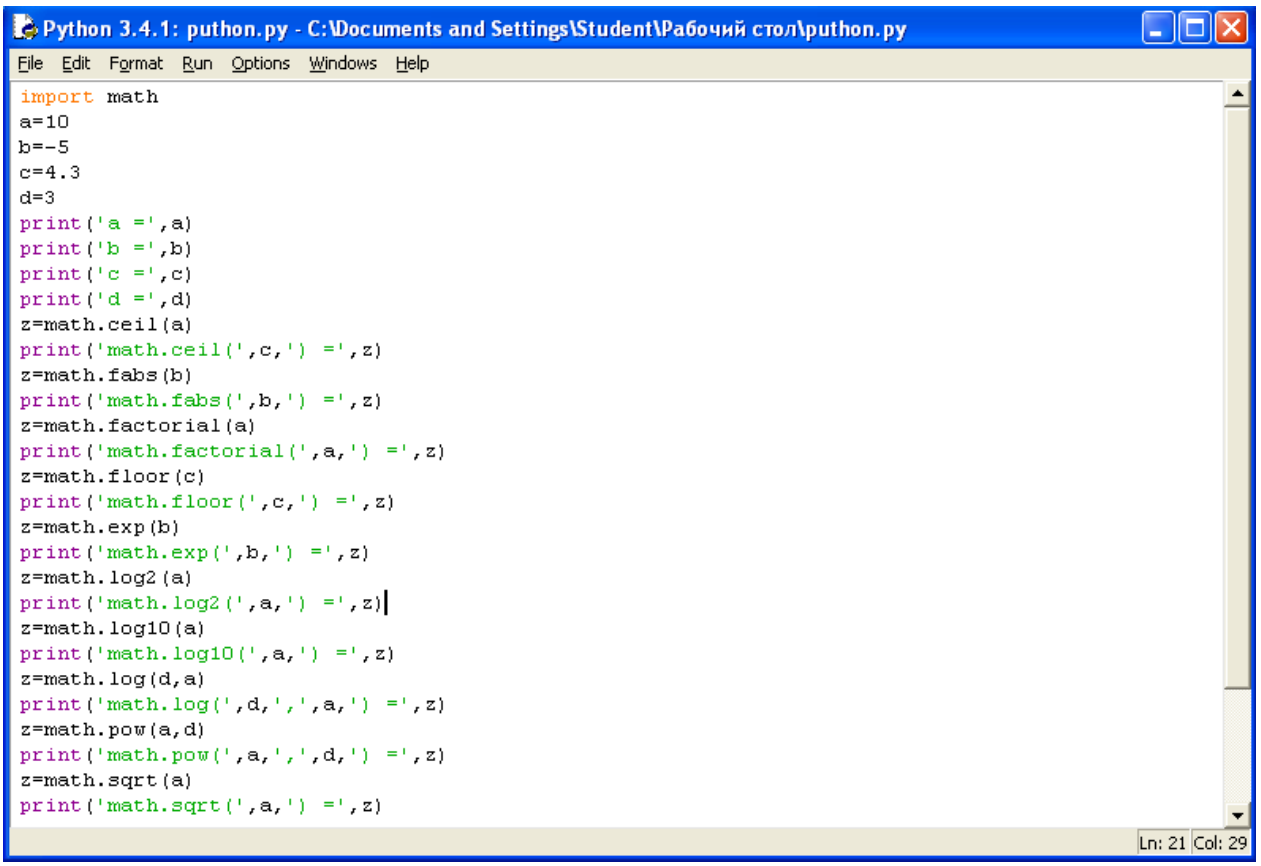


Рассмотрим наиболее часто используемые функции модуля math

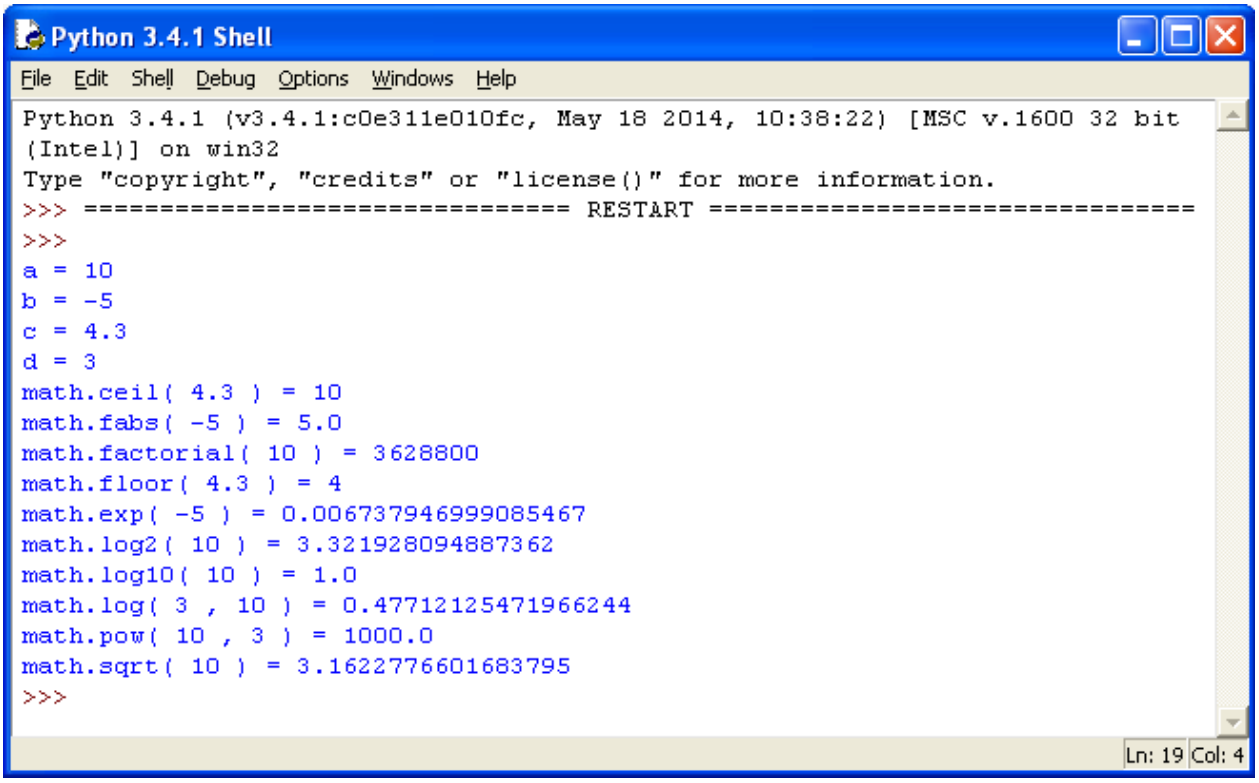
math.ceil(x)	Возвращает ближайшее целое число большее, чем x
math.fabs(x)	Возвращает абсолютное значение числа x
math.factorial(x)	Вычисляет факториал x
math.floor(x)	Возвращает ближайшее целое число меньшее, чем x
math.exp(x)	Вычисляет e**x
math.log2(x)	Логарифм по основанию 2
math.log10(x)	Логарифм по основанию 10
math.log (x[, base])	По умолчанию вычисляет логарифм по основанию e, дополнительно можно указать основание логарифма
math.pow(x, y)	Вычисляет значение x в степени y
math.sqrt(x)	Корень квадратный от x

Пример применения вышеописанных функций над числами

В программе определены 4 переменные - a, b, c, d, каждая из которых является либо целым числом, либо вещественным, либо отрицательным.Командой print() выводится значение каждой переменной на экран при выполнении программы.В переменную z помещается результат выполнения функции модуля math. Затем командой print() выводится сообщение в виде используемой функции и её аргумента и результат её выполнения.



Пример программы на Python



Результат выполнения программы с применением функций модуля math

Тригонометрические функции модуля math

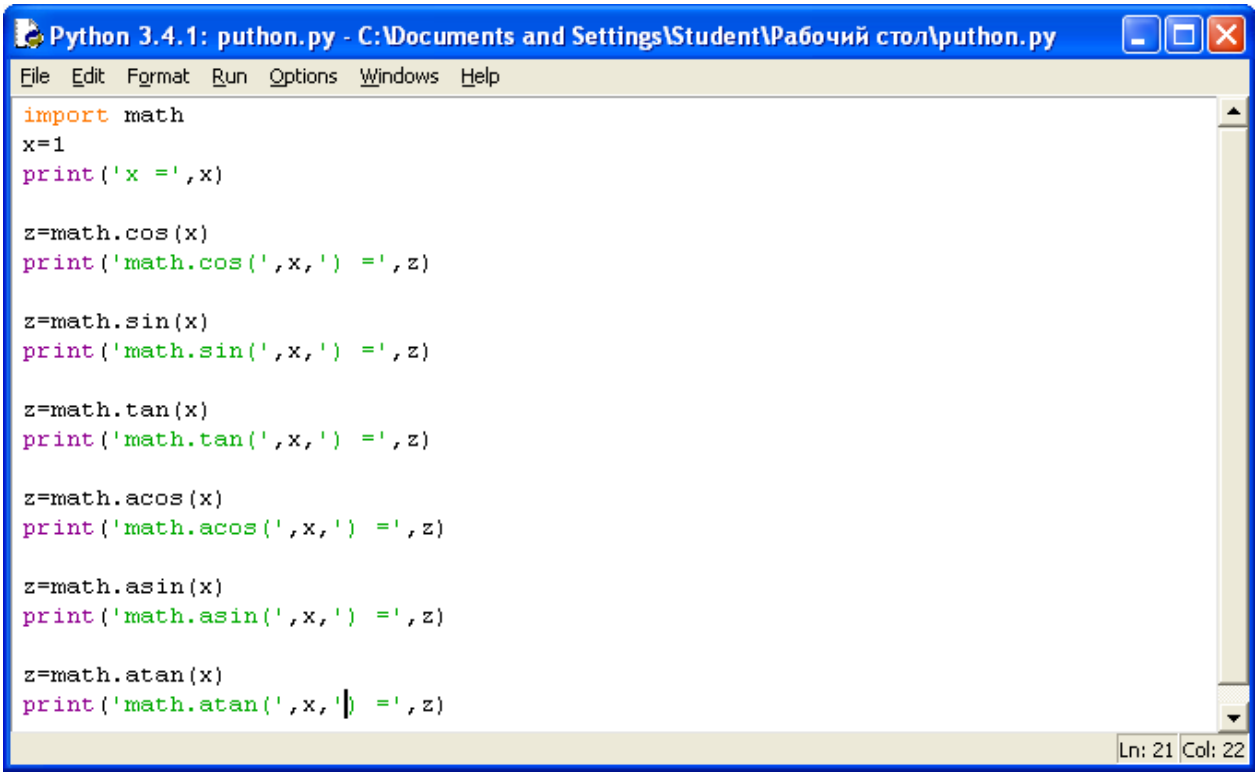
math.cos(x)	Возвращает cos числа X
math.sin(x)	Возвращает sin числа X
math.tan(x)	Возвращает tan числа X
math.acos(x)	Возвращает acos числа X
math.asin(x)	Возвращает asin числа X
math.atan(x)	Возвращает atan числа X

Пример применения вышеописанных функций над числами

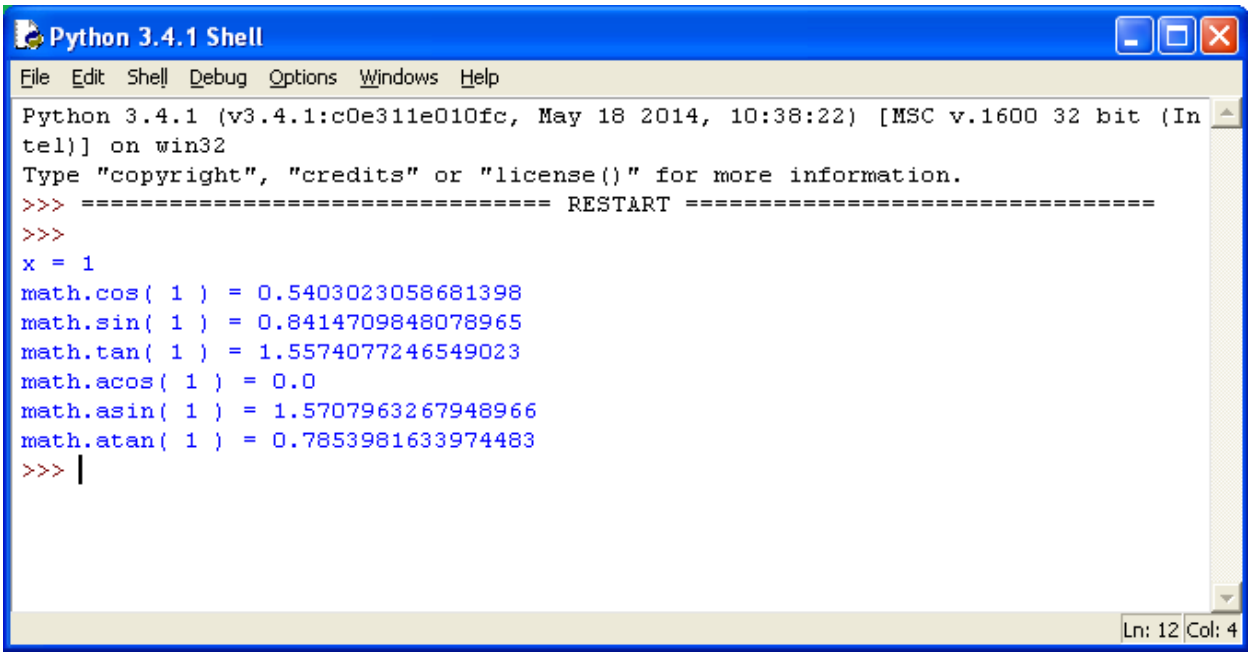
В программе определена переменная x, содержащая целое число. Значение переменной выводится командой print() на экран.

В переменную z помещается результат выполнения тригонометрической функции модуля math.

Затем командой print() выводится сообщение в виде используемой функции и её аргумента и результат выполнения.



Пример программы с использованием тригонометрических функций модуля math



Результат выполнения программы с применением тригонометрических функций модуля math

Константы:

- **math.pi** - число Pi.
- **math.e** - число e (экспонента).

Пример

Напишите программу, которая бы вычисляла заданное арифметическое выражение при заданных переменных. Ввод переменных осуществляется с клавиатуры. Вывести результат с 2-мя знаками после запятой.

Задание

$$Z = \frac{9\pi t + 10\cos(x)}{\sqrt{t} - |\sin(t)|} * e^{x}$$

x=10;
t=1

Решение

Сначала импортируем модуль math. Для этого воспользуемся командой import math.

Затем следует ввести значения двух переменных целого типа x и t.

Для ввода данных используется команда input, но так как в условии даны целые числа, то нужно сначала определить тип переменных: x=int(), t=int().

Определив тип переменных, следует их ввести, для этого в скобках команды int() нужно написать команду input().

Для переменной x это выглядит так: x=int(input("сообщение при вводе значения")).

Для переменной t аналогично: t=int(input("сообщение при вводе значения")).
Следующий шаг - это составление арифметического выражения, результат которого поместим в переменную z.

Сначала составим числитель. Выглядеть он будет так:
9*math.pi*t+10*math.cos(x).

Затем нужно составить знаменатель, при этом обратим внимание на то, что числитель делится на знаменатель, поэтому и числитель и знаменатель нужно поместить в скобки (), а между ними написать знак деления /.

Выглядеть это будет так: (9*math.pi*t+10*math.cos(x))/(math.sqrt(t)-math.fabs(math.sin(t))).

Последним шагом является умножение дроби на экспоненту в степени x.

Так как умножается вся дробь, то следует составленное выражение поместить в скобки (), а уже потом написать функцию math.pow(math.e,x).

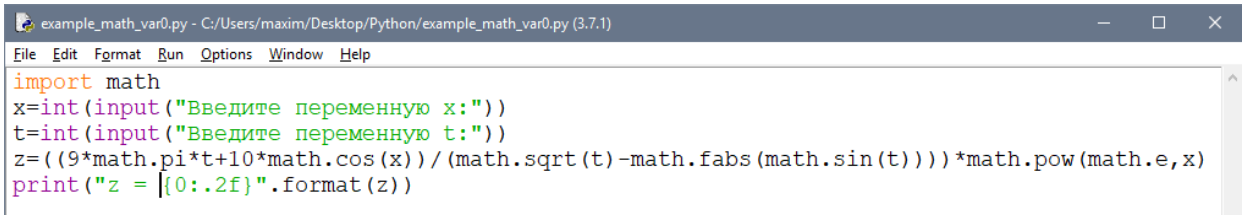
В результате выражение будет иметь вид:
z=((9*math.pi*t+10*math.cos(x))/(math.sqrt(t)-
math.fabs(math.sin(t))))*math.pow(math.e,x).

При составлении данного выражения следует обратить внимание на количество открывающихся и закрывающихся скобок.

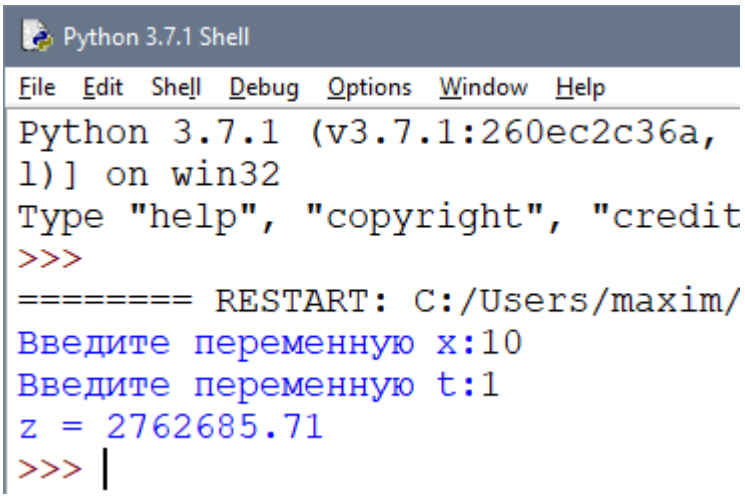
Командой print() выведем значение переменной, отформатировав его командой format.

Сам формат записывается в апострофах в фигурных скобках {}.

В задаче требуется вывести число с двумя знаками после запятой, значит вид формата будет выглядеть следующим образом: {0:.2f}, где 2 - это количество знаков после запятой, а f указывает на то, что форматируется вещественное число. При этом перед 2 нужно поставить точку, указав тем самым на то, что форматируем именно дробную часть числа.



Результат



Практическая работа 6

Условный оператор. Структура ветвление в Python. Работа с циклами в Python

1. Цель работы: познакомиться со структурой ветвление (if, if-else, if-elif-else). Научиться работать с числами и строками используя данную структуру, познакомиться с циклическими конструкциями.

2. Задачи работы:

- познакомиться со структурой ветвления в Python
- познакомиться с циклическими структурами (while и for)

В соответствии общеобразовательной учебной дисциплиной «Информатика» в результате выполнения заданий ЛР, студент должен:

иметь представление:

- о принципах кодирования информации; о системах счисления;

знать:

- различные подходы к определению понятия «информация»;
- методы измерения количества информации: вероятностный и алфавитный. Знать единицы измерения информации;

уметь:

- оценивать достоверность информации, сопоставляя различные источники;
- использовать готовые информационные модели, оценивать их соответствие реальному объекту и целям моделирования;
- осуществлять выбор способа представления информации в соответствии с поставленной задачей;
- представлять числовую информацию различными способами (таблица, массив, график, диаграмма и пр.).

3. Подготовка к работе

Ознакомиться с теоретической справкой

4. Задание

1) Выполнить задание в соответствии с вариантом из приложения 2
2) Выполнить задание в соответствии с вариантом из предложения 3

5. Порядок выполнения работы

1. Ознакомиться с теоретической справкой
2. Выполнить задание 1
3. Выполнить задание 2
4. Представить преподавателю отчет

6. Содержание отчёта

1. Название, цель работы
2. Выполнение п.5 ()
3. Выводы по работе.

Приложение

Теоретическая справка
(часть 1)

Условный оператор ветвления if, if-else, if-elif-else
Оператор ветвления if позволяет выполнить определенный набор инструкций в зависимости от некоторого условия. Возможны следующие варианты использования.

1. Конструкция if
Синтаксис оператора if выглядит так:
if логическое выражение:
    команда_1
    команда_2
    ...
    команда_n

После оператора if записывается логическое выражение.
Логическое выражение — конструкция языка программирования, результатом вычисления которой является «истина» или «ложь». Если это выражение истинно, то выполняются инструкции, определяемые данным оператором. Выражение является истинным, если его результатом является число не равное нулю, непустой объект, либо логическое True. После выражения нужно поставить двоеточие ":".

ВАЖНО: блок кода, который необходимо выполнить, в случае истинности выражения, отделяется четырьмя пробелами слева! Программа запрашивает у пользователя два числа, затем сравнивает их и если числа равны, то есть логическое выражение A==B истинно, то выводится соответствующее сообщение.

2. Конструкция if – else
Бывают случаи, когда необходимо предусмотреть альтернативный вариант выполнения программы. Т.е. при истинном условии нужно выполнить один набор инструкций, при ложном – другой. Для этого используется конструкция if – else.
Синтаксис оператора if – else выглядит так:
if логическое выражение:
    команда_1
    команда_2
    ...

    команда_n
else:
    команда_1
    команда_2
    ...
    команда_n
Программа запрашивает у пользователя два числа, затем сравнивает их и если числа равны, то есть логическое выражение A==B истинно, то выводится соответствующее сообщение. В противном случае выводится сообщение, что числа не равны.

3. Конструкция if – elif – else
Для реализации выбора из нескольких альтернатив можно использовать конструкцию if – elif – else. Синтаксис оператора if – elif – else выглядит так:
if логическое выражение_1:

```
        команда_1
        команда_2
        ...
        команда_n
elif логическое выражение_2:
        команда_1
        команда_2
        ...
        команда_n
elif логическое выражение_3:
        команда_1
        команда_2
        ...
        команда_n
else:
        команда_1
        команда_2
        ...
        команда_n
```

Программа запрашивает число у пользователя и сравнивает его с нулём a<0. Если оно меньше нуля, то выводится сообщение об этом. Если первое логическое выражение не истинно, то программа переходит ко второму - a==0. Если оно истинно, то программа выведет сообщение, что число равно нулю, в противном случае, если оба вышеуказанных логических выражения оказались ложными, то программа выведет сообщение, что введённое число больше нуля.

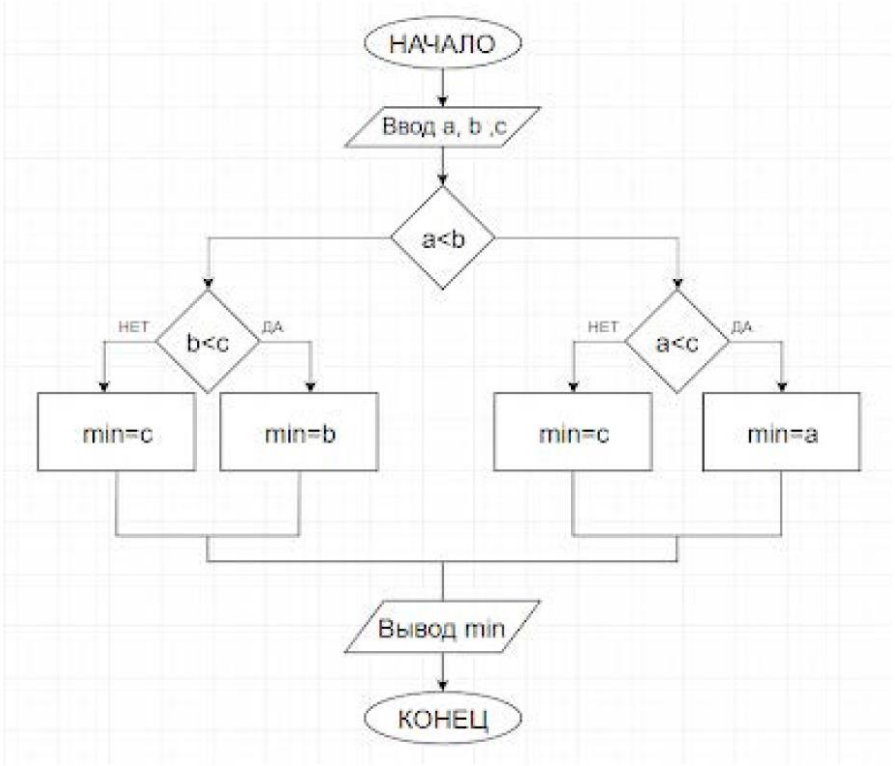
Пример

Вариант 0

Дано 3 числа. Найти минимальное среди них и вывести на экран.

Решение

Для простоты построим блок-схему задачи.



Командами
a=input('')
b=input('')
c=input('')
введём три числа, присвоив значения переменным a, b, c.
Условной конструкцией if-else проверим на истинность логическое выражение a<b. Если оно истинно, то переходим на проверку логического выражения a<c. Если оно истинно, то переменной "y" присвоим значение переменной "a", т.е. "a" будет минимальным, а иначе "y" присвоится значение переменной "c". Если в начале логическое выражение a<b оказалось ложным, то переходим на проверку другого логического выражения b<c. Если оно истинно, то "y" присвоится значение переменной "b", иначе "c". Командой print() выводим минимальное значение.

Теоретическая справка
(часть 2)

1. Цикл while в Python

Инструкция while в Python повторяет указанный блок кода до тех пор, пока указанное в цикле логическое выражение будет оставаться истинным.

Синтаксис цикла while:

while логическое выражение:

команда 1
команда 2
...
команда n

После ключевого слова while указывается условное выражение, и пока это выражение возвращает значение True, будет выполняться блок инструкций, который идет далее.
Все инструкции, которые относятся к циклу while, располагаются на последующих строках и должны иметь отступ от начала строки (4 пробела).

```
#! Программа по вычислению факториала
number = int(input("Введите число: "))
i = 1
factorial = 1
while i <= number:
    factorial *= i
    i += 1
print("Факториал числа", number, "равен", factorial)
```

Пример программы на Python

```
Python 3.4.1 (v3.4.1:c0e311e010fc, May 18 2014, 10:38:22) [MSC v.1600 32 bit (In
tel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> ================================ RESTART ================================
>>>
Введите число: 5
Факториал числа 5 равен 120
>>>
```

Результат выполнения программы с использованием циклического оператора while

2.Цикл for в Python:

Цикл for в Python обладает способностью перебирать элементы любого комплексного типа данных (например, строки или списка).
Синтаксис цикла for:

for int in range():

команда 1
команда 2
...
команда n

Переменной int присваивается значение первого элемента функции range(), после чего выполняются команды. Затем переменной int присваивается следующее по порядку значение и так далее до тех пор, пока не будут перебраны все элементы функции range(). Функция range() является универсальной функцией Python для создания списков (list) содержащих арифметическую прогрессию. Чаще всего она используется в циклах for. range(старт, стоп, шаг) - так выглядит стандартный вызов функции range() в Python. По умолчанию старт равняется нулю, шаг единице.

Вариант 0

1. Найти сумму n элементов следующего ряда чисел: 1 -0.5 0.25 -0.125 ... n. Количество элементов (n) вводится с клавиатуры. Вывести на экран каждый член ряда и его сумму. Решить задачу используя циклическую конструкцию for.

Решение:
В данном случае ряд чисел состоит из элементов, где каждый следующий меньше предыдущего в два раза по модулю и имеет обратный знак. Значит, чтобы получить следующий элемент, надо предыдущий разделить на -2.

Какой-либо переменной надо присвоить значение первого элемента ряда (в данном случае это 1). Далее в цикле добавлять ее значение к переменной, в которой накапливается сумма, после чего присваивать ей значение следующего элемента ряда, разделив текущее значение на -2. Цикл должен выполняться n раз.

```
n=int(input('Введите количество элементов последоваетльности: '))
x=1
s=0
print(x)
for i in range(n):
    s+=x
    x/=-2
    print(x)
print('Сумма ряда:',s)
```

Пример программы с циклом for

```
Введите количество элементов последоваетльности: 5
1
-0.5
0.25
-0.125
0.0625
Сумма ряда: 0.6875
```

Результат выполнения программы

2. Дано целое число, не меньшее 2. Выведите его наименьший натуральный делитель, отличный от 1.

Решение:
Для начала введём целое число командой int(input(текст сообщения)).
Затем зададим переменной i значение 2. Переменная i выполняет роль счётчика. Если задать ей значение 1, то условие задачи не будет выполнено, а результатом всегда будет 1.
В цикле while в качестве логического выражения используется сравнение остатка от деления n на i с нулём. Таким образом, если остаток от деления введённого числа на текущее значение i не равно нулю, то счётчик увеличивается на 1, а если равно нулю цикл заканчивается и командой print() выводится сообщение и значение i.

```
n = int(input('Введите целое число не меньшее 2\n'))
i = 2
while n%i != 0:
    i+=1
print('наименьший натуральный делитель:',i)
```

Пример программы с циклом while

```
Введите целое число не меньшее 2
49
наименьший натуральный делитель: 7
```

Результат выполнения программы

Приложение 2

Задание 1

Вариант 1

Даны три целых числа. Выбрать из них те, которые принадлежат интервалу [1,3].

Вариант 2

Дан номер года (положительное целое число). Определить количество дней в этом году, учитывая, что обычный год насчитывает 365 дней, а високосный — 366 дней. Високосным считается год, делящийся на 4, за исключением тех годов, которые делятся на 100 и не делятся на 400 (например, годы 300, 1300 и 1900 не являются високосными, а 1200 и 2000 — являются).

Вариант 3

Написать программу вычисления стоимости покупки с учетом скидки. Скидка в 3% предоставляется в том случае, если сумма покупки больше 500 руб., в 5% - если сумма больше 1000 руб.

Вариант 4

Написать программу, которая бы по введенному номеру единицы измерения (1 — килограмм, 2 — миллиграмм, 3 — грамм, 4 — тонна, 5 — центнер) и массе M выдавала соответствующее значение массы в килограммах.

Вариант 5

Найти косинус минимального из 4 заданных чисел.

Вариант 6

Вывести на экран синус максимального из 3 заданных чисел.

Вариант 7

Даны три стороны одного треугольника и три стороны другого треугольника. Определить, будут ли эти треугольники равновеликими, т. е. имеют ли они равные площади. Если это не так, то вывести «Foul!!!»

Вариант 8

Составьте программу подсчета площади равнобедренного треугольника. Если площадь треугольника чётная, разделить её на 2, в противном случае вывести сообщение «Не могу делить на 2!»

Вариант 9

Составить программу, которая по данному числу (1-12) выводит название соответствующего ему месяца на английском языке.

Вариант 10

Составить программу, осуществляющую перевод величин из радианной меры в градусную или наоборот. Программа должна запрашивать, какой перевод нужно осуществить, и выполнять указанное действие.

Вариант 11

Дано три числа. Найти количество положительных чисел среди них;

Вариант 12

Если действительные числа x и y – одного знака, найти их среднее геометрическое, в противном случае найти их среднее арифметическое.

Вариант 13

Определить, существует ли прямоугольный треугольник со сторонами x,y,z. Если – да, вычислить его площадь.

Вариант 14

Определить, существует ли треугольник с длинами сторон a, b, c. Если – да, вычислить его площадь по формуле Герона.
Формула Герона имеет вид:
S=p(p-a)(p-b)(p-c), где p=12(a+b+c)

Вариант 15

Вычислить значение функции f(x), если

$$f(x)=\begin{cases} 0{,}5-\sqrt[4]{|x|}, & x\geq 0 \\ \dfrac{\sin^2 x^2}{|x+1|}, & x<0 \end{cases}$$

Приложение 3

Задание 2

Вариант 1

1. Дано вещественное число – цена 1 кг конфет. Вывести стоимость 1, 2, ... 10 кг конфет. Решить задачу используя циклическую конструкцию for.
2. Дана непустая последовательность целых чисел, оканчивающаяся нулем. Найти: а) сумму всех чисел последовательности; б) количество всех чисел последовательности.
Решить задачу используя циклическую конструкцию while.

Вариант 2

1. Даны два числа A и B (A<B). Найти сумму всех целых чисел от A до B включительно. Решить задачу используя циклическую конструкцию for.
2. Дана последовательность отрицательных целых чисел, оканчивающаяся положительным числом. Найти среднее арифметическое всех чисел последовательности (без учета положительным числа).
Решить задачу используя циклическую конструкцию while.

Вариант 3

1. Даны два числа A и B (A<B). Найти сумму квадратов всех целых чисел от A до B включительно. Решить задачу используя циклическую конструкцию for.
2. Дана последовательность из n целых чисел. Первое число в последовательности чётное. Найти сумму всех идущих подряд в начале последовательности чётных чисел. Условный оператор не использовать Решить задачу используя циклическую конструкцию while.

Вариант 4

1. Найти среднее арифметическое всех целых чисел от a до 200 (значения a и b вводятся с клавиатуры; a≤ 200). Решить задачу используя циклическую конструкцию for.
2. Дана последовательность из n вещественных чисел, начинающаяся с положительного числа. Определить, какое количество положительных чисел записано в начале последовательности. Условный оператор не использовать. Решить задачу используя циклическую конструкцию while.

Вариант 5

1. Найти сумму всех целых чисел от a до b (значения a и b вводятся с клавиатуры; b ≥a). Решить задачу используя циклическую конструкцию for.
2.Дано целое число N (> 0), являющееся некоторой степенью числа 2: N = 2^K. Найти целое число K — показатель этой степени.
Решить задачу используя циклическую конструкцию while.

Вариант 6

1. Найти сумму квадратов всех целых чисел от a до 50 (значение a вводится с клавиатуры; 0 ≤a≤50). Решить задачу используя циклическую конструкцию for.
2.Дано целое число N (> 1). Найти наименьшее целое число K, при котором выполняется неравенство 5^K > N.
Решить задачу используя циклическую конструкцию while.

Вариант 7

1. Дана непустая последовательность целых чисел, оканчивающаяся нулем.
Найти:
а) сумму всех чисел последовательности;
б) количество всех чисел последовательности.
Решить задачу используя циклическую конструкцию for.
2.Дано целое число N (> 1). Найти наибольшее целое число K, при котором выполняется неравенство 2^K > N.
Решить задачу используя циклическую конструкцию while.

Вариант 8

1. Дана последовательность из n вещественных чисел. Первое число в последовательности нечетное. Найти сумму всех идущих подряд в начале последовательности нечетных чисел. Условный оператор не использовать. Решить задачу используя циклическую конструкцию for.
2.Дано целое число N (> 0). Используя операции деления нацело и взятия остатка от деления, найти количество и сумму его цифр.
Решить задачу используя циклическую конструкцию while.

Вариант 9

1. Среди чисел 1, 4, 9, 16, 25, ... найти первое число, большее n. Решить задачу используя циклическую конструкцию for.
2.Среди чисел 1, 5 10, 16, 23, ... найти первое число, большее n. Условный оператор не использовать.
Решить задачу используя циклическую конструкцию while.

Вариант 10

1. Известны оценки по физике каждого из 20 учеников класса. Определить среднюю оценку.
Решить задачу используя циклическую конструкцию for.
2.Дано число A (> 1). Вывести наибольшее из целых чисел K, для которых сумма 1 + 1/2 + ... + 1/K будет меньше A, и саму эту сумму.
Решить задачу используя циклическую конструкцию while.

Вариант 11

1. Известно сопротивление каждого из элементов электрической цепи. Все элементы соединены последовательно. Определить общее сопротивление цепи.
Решить задачу используя циклическую конструкцию for.
2.Дано целое число N (> 0). Найти наибольшее целое число K, квадрат которого не превосходит N: K^2 ≤ N. Функцию извлечения квадратного корня не использовать.
Решить задачу используя циклическую конструкцию while.

Вариант 12

1. Известны оценки по физике каждого ученика двух классов. Определить среднюю оценку в каждом классе. Количество учащихся в каждом классе одинаковое.
Решить задачу используя циклическую конструкцию for.
2. Выведите на экран для числа 2 его степени от 0 до 20
Решить задачу используя циклическую конструкцию while.

Вариант 13

1. В области 12 районов. Известны количество жителей (в тысячах человек) и площадь (в км2) каждого района. Определить среднюю плотность населения по области в целом.
Решить задачу используя циклическую конструкцию for.
2.Мой богатый дядюшка подарил мне один доллар в мой первый день рождения. В каждый день рождения он удваивал свой подарок и прибавлял к нему столько долларов, сколько лет мне исполнилось. Написать программу, указывающую, к какому дню рождения подарок превысит 100$.
Решить задачу используя циклическую конструкцию while.

Вариант 14

1. Одноклеточная амеба каждые 3 часа делится на 2 клетки. Определить, сколько клеток будет через 3, 6, 9, ..., 24 часа, если первоначально была одна амеба.
Решить задачу используя циклическую конструкцию for.
2. Вывести таблицу значений функции y = -0.5x + x. Значения аргумента (x) задаются минимумом, максимумом и шагом. Например, если минимум задан как 1, максимум равен 3, а шаг 0.5. То надо вывести на экран изменение x от 1 до 3 с шагом 0.5 (1, 1.5, 2, 2.5, 3) и значения функции (y) при каждом значении x.
Решить задачу используя циклическую конструкцию while.

Вариант 15

1. Начав тренировки, лыжник в первый день пробежал 10 км. Каждый следующий день он увеличивал пробег на 10% от пробега предыдущего дня. Определить:

а) пробег лыжника за второй, третий, ..., десятый день тренировок;

б) какой суммарный путь он пробежал за первые 7 дней тренировок.

Решить задачу используя циклическую конструкцию for.

2. Найти сумму и произведение цифр, введенного целого числа. Например, если введено число 325, то сумма его цифр равна 10 (3+2+5), а произведение 30 (3*2*5).

Решить задачу используя циклическую конструкцию while.

Практическая работа 7
Использование поисковых серверов

1. **Цель работы**: ознакомиться с работой в поисковой системы
2. **Задачи работы:**
   - ознакомиться с работой в поисковой системы
   - ознакомиться с особенностями поисков по ключевым словам, тегам
   - изучить особенности поисковых систем в отношении поиска информации.

В соответствии общеобразовательной учебной дисциплиной «Информатика» в результате выполнения заданий ПР, студент должен:

иметь представление:
- о методах поиска информации;
- о возможности соединения разнотипной информации в одном электронном документе с помощью технологии мультимедиа;

знать:
- различные подходы к определению понятия «информация»;
- назначение наиболее распространенных средств автоматизации информационной деятельности (текстовых редакторов, текстовых процессоров, графических редакторов, электронных таблиц, баз данных, компьютерных сетей);
- назначение и виды информационных моделей, описывающих реальные объекты или процессы;

уметь:
- оценивать достоверность информации, сопоставляя различные источники;
- осуществлять выбор способа представления информации в соответствии с поставленной задачей;
- иллюстрировать учебные работы с использованием средств информационных технологий;
- представлять числовую информацию различными способами (таблица, массив, график, диаграмма и пр.);
- соблюдать правила техники безопасности и гигиенические рекомендации при использовании средств ИКТ.

3. **Подготовка к работе**

Для обеспечения выполнения работы необходимо иметь компьютер со следующим обеспечением: операционная система Windows, браузер, подключение к сети “Интернет”.

Подготовить бланк отчета.

4. **Задание**
   1) Ознакомиться с теоретической частью
   2) Выполнить поисковые запросы в соответствии с пунктом 5
5. **Порядок выполнения работы**

Поиск по сервисам поисковой системы

1). Запустите браузер, введите в строку адреса yandex.ru

Способ поиска по рубрикам поискового сервиса является достаточно быстрым и эффективным. Вам предлагается несколько ссылок, среди которых есть ссылки на нужный Вам материал.

Чтобы ознакомиться со всеми рубриками, нажмите «ещё».

2) Предположим, вы готовите мероприятие ко Дню победы и хотите найти в Интернете известную военную песню Булата Окуджавы «Вы слышите, грохочут сапоги». Вам надо зайти в раздел рубрикатора Музыка и найти нужную песню.

3) Предположим, вы собираетесь приобрести мобильный телефон и хотите сравнить характеристики аппаратов разных фирм.

Организуйте поиск по следующим рубрикам каталога: Яндекс > Маркет > Мобильные телефоны.

Поиск по ключевым словам

Предположим, что мы решили завести аквариум и нас интересует любая информация по данной теме. На первый взгляд самое простое — это поиск по слову аквариум.

Введите в строку поиска **аквариум**

Напишите в документ ПР7, что покажет поисковая система

Результатом поиска будет огромное количество страниц - огромное количество ссылок. Причем, если посмотреть внимательнее, среди них окажутся сайты, упоминающие группу Б. Гребенщикова «Аквариум», торговые центры и неформальные объединения с таким же названием, и многое другое, не имеющее отношения к аквариумным рыбкам.

Вести поиск по одному слову, как правило, нецелесообразно, ведь по одному слову очень сложно определить тему, которой посвящен документ, Web-страница или сайт. Исключение составляют редкие слова и термины, которые практически никогда не используются вне своей тематической области.

Имея определенный набор наиболее употребительных терминов в нужной области, можно использовать расширенный поиск. В этом режиме возможности языка запросов реализованы в виде формы. Подобный сервис, включающий словарные фильтры, предлагается почти всеми поисковыми системами.

Но мы опробуем уточнить условия поиска, используя язык запросов.

Введите в строку поиска словосочетание аквариумные рыбки

Напишите документ ПР7, что покажет поисковая система

Количество ссылок уменьшится и среди них на первых страницах не будет ссылок на сайты, не имеющих отношения к теме поиска.

Этот результат нас устраивает больше, но все равно среди предложенных ссылок могут встретиться, например, русские сувенирные наборы спичечных этикеток с изображениями рыбок, и коллекции заставок для Рабочего стола компьютера, и каталоги аквариумных рыбок с фотографиями, и магазины аквариумных аксессуаров. Очевидно, что следует продолжить движение в направлении уточнения условий поиска.

Для того чтобы сделать поиск более продуктивным, во всех поисковых системах существует специальный язык формирования запросов со своим синтаксисом. Эти языки во многом похожи. Изучить их все достаточно сложно, но любая поисковая машина имеет справочную систему, которая позволит вам освоить нужный язык.

Правила формирования запросов в поисковой системе

Изучите правила формирования запросов в Яндексе, используя Яндекс.Помощь.

Наберите в поисковой строке «Яндекс.помощь».

Или использовать Справку по сервисам, прокрутив страницу Яндекс.помощи вниз и выбрав нужный сервис.

На открывшейся странице выберите «Язык запросов».

Затем выберите «Морфология и поисковый контекст».

Морфология и поисковый контекст

При поиске с учетом морфологии принимаются во внимание:

форма заданного слова (падеж, род, число, склонение и т. д.);

часть речи (существительное, прилагательное, глагол и т. д.).

По умолчанию Яндекс ищет все формы слова, указанного в запросе. Например, при запросе рассказал поиск будет производиться по глагольным формам «рассказать», «расскажу», «рассказывать» и т. д., но не по однокоренным словам типа «рассказ», «рассказчик». Исключение составляют случаи, когда используются операторы ! и " .

Также вы можете конкретизировать поисковый запрос с помощью операторов, которые уточняют наличие запрашиваемых слов в документе.

6. Содержание отчёта

1. Название, цель работы
2. Выполнение п.5
3. Выводы по работе.

Приложение

Краткие сведения из теории

Для поиска интересующей вас информации необходимо указать адрес Web-страницы, на которой она находится. Это самый быстрый и надежный вид поиска. Адреса Web-страниц приводятся в специальных справочниках, печатных изданиях, звучат в эфире популярных радиостанций и с экранов телевизора.

Поисковые системы

Если вы не знаете адреса, то для поиска информации в сети Интернет существуют поисковые системы, которые содержат информацию о ресурсах Интернета.

Каждая поисковая система – это большая база ключевых слов, связанных с Web-страницами, на которых они встретились. Для поиска адреса сервера с интересующей вас информацией надо ввести в поле поисковой системы ключевое слово, несколько слов или фразу. Тем самым вы посылаете поисковой системе запрос. Результаты поиска выдаются в виде списка адресов Web-страниц, на которых встретились эти слова.

Как правило, поисковые системы состоят из трех частей: робота, индекса и программы обработки запроса.

Робот (Spider, Robot или Bot) - это программа, которая посещает Web-страницы и считывает (полностью или частично) их содержимое. Роботы поисковых систем различаются индивидуальной схемой анализа содержимого Web -страницы.

Индекс - это хранилище данных, в котором сосредоточены копии всех посещенных роботами страниц. Индексы в каждой поисковой системе различаются по объему и способу организации хранимой информации. Базы данных ведущих поисковых машин хранят сведения о десятках миллионов документов, а объемы их индекса составляют сотни гигабайт. Индексы периодически обновляются и дополняются, поэтому результаты работы одной поисковой машины с одним и тем же запросом могут различаться, если поиск производился в разное время.

Программа обработки запроса - это программа, которая в соответствии с запросом пользователя «просматривает» индекс на предмет наличия нужной информации и возвращает ссылки на найденные документы.

Наиболее распространенными поисковыми системами являются:

Яндекс (www.yandex.ru)

Гугл (www.google.ru)

Рамблер (www.rambler.ru)

Результаты поиска выстраиваются по значимости – наиболее важные документы размещаются в начале списка. При этом положение найденного документа в списке определяется тем, в каком месте документа находится ключевое слово (в заглавии документа важнее, чем в любом другом месте) и числом упоминаний ключевого слова (чем больше упоминаний, тем ранг выше).

Таким образом, сайты, расположенные на первых местах в списке, являются ведущими не с содержательной точки зрения, а практически, по отношению к частоте упоминания ключевого слова. В связи с этим, не следует ограничиваться просмотром первого десятка предложенных поисковой системой сайтов.

Список документов, предлагаемый поисковой системой в ответ на ключевую фразу или слово, может оказаться огромным. В связи с этим в мощных Поисковых Машинах предоставлена возможность в рамках первого списка, выбрать документы, которые точнее отражают цель поиска, то есть уточнить или улучшить результаты поиска, с помощью команды «Искать в найденном».

Поиск по рубрикатору поисковой системы

Поисковые рубрикаторы (каталоги) представляют собой систематизированную коллекцию (подборку) ссылок на ресурсы Интернета. Ссылки организованы в виде тематического рубрикатора, представляющего собой иерархическую структуру, перемещаясь по которой, можно найти нужную информацию.

Это каталог общего назначения, так как в нем представлены ссылки на ресурсы Интернета практически по всем возможным направлениям. В каталоге могут быть выделены темы. Каждая тема включает множество подразделов, а они, в свою очередь, содержат рубрики и т.д. Либо материалы сгруппированы по каким-либо признакам.

Поиск по ключевым словам

Большинство поисковых машин имеют возможность поиска по ключевым словам. Это один из самых распространенных видов поиска.

Для поиска по ключевым словам необходимо ввести в специальном окне слово или несколько слов, которые следует искать, и щелкнуть на кнопке Найти.

Поисковая система найдет в своей базе и покажет документы, содержащие эти слова. Таких документов может оказаться множество, но много в данном случае не обязательно означает хорошо.

Практическая работа 8

Проводная и беспроводная передача информации между компьютерами

1. **Цель работы**: ознакомиться с видами передачи данных между компьютерами
2. **Задачи работы:**
   - ознакомиться с основными устройствами передачи информации между компьютерами
   - ознакомиться с принципами работы основных устройств передачи информации между компьютерами

В соответствии общеобразовательной учебной дисциплиной «Информатика» в результате выполнения заданий ПР, студент должен:

иметь представление:

- об информационных основах процессов управления;
- о возможности соединения разнотипной информации в одном электронном документе с помощью технологии мультимедиа;

знать:

- различные подходы к определению понятия «информация»;
- методы измерения количества информации: вероятностный и алфавитный. Знать единицы измерения информации;
- назначение наиболее распространенных средств автоматизации информационной деятельности (текстовых редакторов, текстовых процессоров, графических редакторов, электронных таблиц, баз данных, компьютерных сетей);
- назначение и виды информационных моделей, описывающих реальные объекты или процессы;

уметь:

- оценивать достоверность информации, сопоставляя различные источники;
- распознавать информационные процессы в различных системах;
- использовать готовые информационные модели, оценивать их соответствие реальному объекту и целям моделирования;
- осуществлять выбор способа представления информации в соответствии с поставленной задачей;
- создавать информационные объекты сложной структуры;
- представлять числовую информацию различными способами (таблица, массив, график, диаграмма и пр.);
- соблюдать правила техники безопасности и гигиенические рекомендации при использовании средств ИКТ.

3. **Подготовка к работе**

Ознакомиться с теоретической частью.

Подготовить бланк отчета.

4. **Задание**
   1) Ознакомиться с теоретической частью
   2) Ответить на поставленные вопросы
5. **Порядок выполнения работы**

Ознакомьтесь с теоретической часть и ответьте на вопросы:

Телекоммуникация - это…?

Модем – это…?

Модулятора модема – это…?

Демодулятор модема – это…?

Дуплексный режим передачи данных – это…?

Полудуплексный режим передачи данных – это…?

Скорость модуляции модема– это…?

Пропускная способность канала связи – это…?
Электронная почта – это…?
Формат электронно-почтового Internet-адреса?

6. Содержание отчёта

1. Название, цель работы
2. Выполнение п.5
3. Выводы по работе.

Приложение

Краткие сведения из теории

Есть три основных способа организации межкомпьютерной связи:

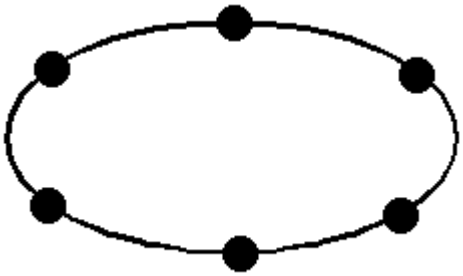
- объединение двух рядом расположенных компьютеров посредством специального кабеля;
- передача данных от одного компьютера к другому посредством модема с помощью проводных, беспроводных или спутниковых линий связи;
- объединение компьютеров в компьютерную сеть

Часто при организации связи между двумя компьютерами за одним компьютером закрепляется роль поставщика ресурсов (программ, данных и т.д.), а за другим — роль пользователя этих ресурсов. В этом случае первый компьютер называется сервером, а второй —клиентом или рабочей станцией. Работать можно только на компьютере-клиенте под управлением специального программного обеспечения.
Сервер (англ. serve — обслуживать) — это высокопроизводительный компьютер с большим объёмом внешней памяти, который обеспечиваетобслуживание других компьютеров путем управления распределением дорогостоящих ресурсов совместного пользования (программ, данных и периферийного оборудования).
Клиент (иначе, рабочая станция) — любой компьютер, имеющий доступ к услугам сервера.
Компьютерная сеть (англ. ComputerNetWork, от net — сеть, и work — работа) — это система обмена информацией между компьютерами.
Пользователи компьютерной сети получают возможность совместно использовать её программные, технические, информационные и организационные ресурсы.
Компьютерная сеть представляет собой совокупность узлов (компьютеров, рабочих станций и др.) и соединяющих их ветвей.
Ветвь сети — это путь, соединяющий два смежных узла.
Узлы сети бывают трёх типов:

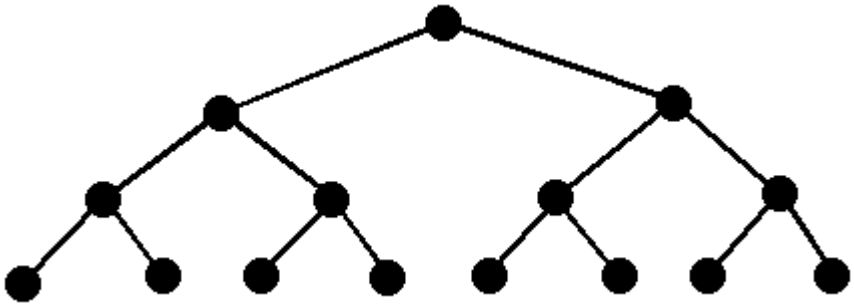
- оконечный узел — расположен в конце только одной ветви;
- промежуточный узел — расположен на концах более чем одной ветви;
- смежный узел — такие узлы соединены по крайней мере одним путём, не содержащим никаких других узлов.

Компьютеры могут объединяться в сеть разными способами. Способ соединения компьютеров в сеть называется её топологией.
Наиболее распространенные виды топологий сетей:
1. Линейная сеть. Содержит только два оконечных узла, любое число промежуточных узлов и имеет только один путь между любыми двумя узлами.

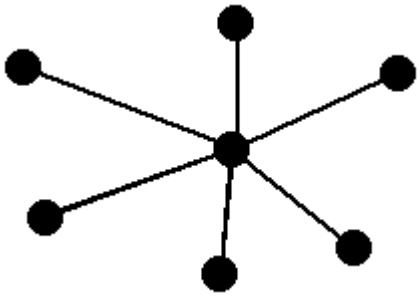
2. Кольцевая сеть. Сеть, в которой к каждому узлу присоединены две и только две ветви.



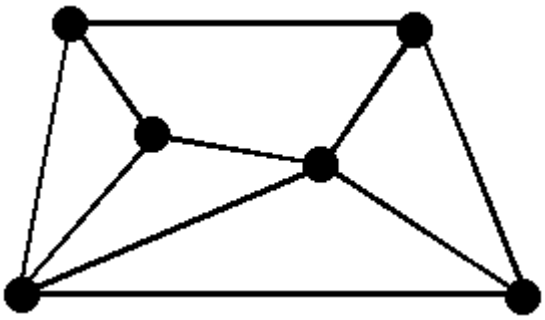
3. Древовидная сеть. Сеть, которая содержит более двух оконечных узлов и по крайней мере два промежуточных узла, и в которой между двумя узлами имеется только один путь.



4. Звездообразная сеть. Сеть, в которой имеется только один промежуточный узел.



5. Ячеистая сеть. Сеть, которая содержит по крайней мере два узла, имеющих два или более пути между ними.



6. Полносвязанная сеть. Сеть, в которой имеется ветвь между любыми двумя узлами. Важнейшая характеристика компьютерной сети — её архитектура.

В современном мире, переживающем информационный бум, всё большее значение приобретает проводная связь - телефония и интернет, которая позволяет людям не только общаться друг с другом на огромном расстоянии, но и пересылать за какие-то доли секунды огромные объёмы информации.

Существует несколько типов проводных линий связи:

1. медная витая пара проводов
2. коаксиальный кабель
3. волоконно-оптическая линия связи

Самой распространённой, дешёвой и простой в монтаже и последующем техническом обслуживании является витая пара. Волоконно-оптическая линия связи, напротив, является наиболее сложной и дорогостоящей.

Несмотря на бурное развитие в последние годы всевозможных средств беспроводной связи, таких, как мобильные или спутниковые телефоны, проводная связь, видимо, будет сохранять свои позиции ещё долгое время.

Основными преимуществами проводной связи перед беспроводной являются простота устройства линий связи и стабильность передаваемого сигнала (качество которого, например, практически не зависит от погодных условий).

Прокладка проводных (кабельных) линий связи для предоставления услуг телефонии и интернет, связана со значительными материальными затратами, а также представляет собой весьма трудоёмкий процесс. Однако, несмотря на подобные сложности, инфраструктура проводной связи постоянно обновляется и совершенствуется.

Беспроводные сетевые технологии группируются в три типа, различающиеся по масштабу действия их радиосистем, но все они с успехом применяются в бизнесе.

1. PAN (персональные сети) — короткодействующие, радиусом до 10 м сети, которые связывают ПК и другие устройства — КПК, мобильные телефоны, принтеры и т. п. С помощью таких сетей реализуется простая синхронизация данных, устраняются проблемы с обилием кабелей в офисах, реализуется простой обмен информацией в небольших рабочих группах. Наиболее перспективный стандарт для PAN — это Bluetooth.
2. WLAN (беспроводные локальные сети) — радиус действия до 100 м. С их помощью реализуется беспроводной доступ к групповым ресурсам в здании, университетском кампусе и т. п. Обычно такие сети используются для продолжения проводных корпоративных локальных сетей. В небольших компаниях WLAN могут полностью заменить проводные соединения. Основной стандарт для WLAN — 802.11.
3. WWAN (беспроводные сети широкого действия) — беспроводная связь, которая обеспечивает мобильным пользователям доступ к их корпоративным сетям и Интернету.

На современном этапе развития сетевых технологий, технология беспроводных сетей Wi-Fi является наиболее удобной в условиях требующих мобильность, простоту установки и использования. Wi-Fi (от англ. wirelessfidelity - беспроводная связь) - стандарт широкополосной беспроводной связи, разработанный в 1997г.

Как правило, технология Wi-Fi используется для организации беспроводных локальных компьютерных сетей, а также создания так называемых горячих точек высокоскоростного доступа в Интернет.