

Лабораторная работа №1.

Линейный вычислительный процесс

Цели и задачи работы: изучение функций ввода-вывода данных, программирования вычисления значения выражения.

Задание к работе: Реализовать линейный вычислительный процесс. Самостоятельно решить задачу в соответствии с индивидуальным вариантом.

Теоретическая часть

При вычислении выражений необходимо анализировать область допустимых значений аргументов, которые используются в выражении. Так, например, знаменатель дроби может получить нулевое значение и программа прервётся по ошибке деления на ноль. Необходимо учитывать и допустимый диапазон аргументов используемых функций. Так, основание логарифма должно быть больше нуля и не равняться единице, а логарифмируемая функция должна быть больше нуля.

Внимательно следует относиться к выражению, в котором, например, выполняется извлечение квадратного корня или, в общем случае, возведение в степень, показатель которой является не целым числом. В этом случае для вычисления используется логарифмирование, и для отрицательного основания степени возникнет ошибка, которая так же приведёт к прерыванию работы программы.

Пример:

Напишите программу для расчета по формулам.

$$1) y = \operatorname{tg}^2\left(\frac{x^2}{2} - 1\right) + \frac{2\cos(x - \pi/6)}{1/2 + \sin^2 \alpha};$$
$$2) y = 2 \frac{\log(3 + \sin(x))(3 - \cos(\pi/4 + 2x))}{1 + \operatorname{tg}^2(2x/\pi)}.$$

Реализация на языке Python:

```
import math

или:

from math import *

import math
print(math.sin(math.pi/4))
print(math.sqrt(2)/2)

или:

from math import *
print(sin(pi/4))

print(sqrt(2)/2)
```

Вместе с тем, такой способ импорта может нарушить пространство имен программы, поскольку может возникнуть конфликт между именами переменных, которые использует программист и именами импортируемых функций.

При импорте можно ограничиться только необходимыми функциями, например:

```
from math import (pi, sin, cos,
                  tan, log)
```

В этом примере демонстрируется способ импорта необходимых функций, и способ размещения инструкции на нескольких строках. Такие функции так же можно использовать в привычной для нас манере.

Больше информации о функциях модуля `math` можно получить из документации или в сети Интернет.

В тех случаях, когда в языке программирования нужная функция отсутствует, ее можно написать, либо вычислить, используя известные формулы. Например,

$$\operatorname{ctg}(x) = 1/\tan(x) = \cos(x)/\sin(x).$$

Имена переменных следует выбирать тщательнее и использовать либо принятые в математике или физике символы, либо фразы, отражающие назначение переменной.

Ввод данных

Ввод данных можно выполнить с клавиатуры функцией `input()`:

```
m = input([str])
```

При этом на экран будет выведена строка `str`, а переменная `m` получит значение строкового типа, введенное пользователем. Строковый тип может быть преобразован, например, к типу `int` или `float`, если введенное значение - число.

Для ввода нескольких значений можно воспользоваться методом `split()`, который позволяет разбить строку на подстроки. Например, для ввода значений параметра α и переменной x можно поступить так:

```
a, x = input('Введите данные (a, x): ').split()
a = float(a)
x = float(x)
```

Используемый разделитель указывается в качестве параметра метода `split()`. Если разделитель не указан, то им будет пробел. При вводе десятичного числа целая часть отделяется от дроби точкой.

Если пользователь не ввел данные (просто нажал Enter) или вместо цифр и точки ввел недопустимые символы, например, буквы, программа завершится аварийно. Исключительная ситуация, которая при этом возникает, может быть обработана с помощью инструкции `try`. Более подробно об этом следует прочитать, например, в [1].

В следующем примере пользователь должен ввести первое число целого типа, а второе – вещественного. Если ввод будет неправильным, то возникнет исключительная ситуация и управление программой будет передано в блок `except`. В этом блоке можно предусмотреть возможные ситуации и принять необходимое решение, например, заставить пользователя правильно ввести данные.

```
import sys, traceback

while True:
    try:
        a = x = ""
        a, x = input('Ввод данных (a: int, x: float): ')\
            .split()
        a = int(a)
        x = float(x)
    except ValueError:
        if a == "" and x == "":
            a = x = 0
            break
    print("a - int, x - float. Пример: 3 4.5")
    print(a, x,)
```

В теле "вечного" цикла, в блоке `try`, инициализируются две переменные, а затем следует инструкция для ввода данных. При ошибочном вводе числа, например, вместо целого – вещественное, или вместо цифры – буква, возникнет исключительная ситуация `ValueError`. Управление будет передано в модуль `except`, где, в условном операторе, проверяется, были ли введены данные. Если был нажат Enter, то переменные получают нулевое значение и управление будет передано инструкции, которая следует за циклом.

Если ввод данных был сделан, то исключительная ситуация возникла при преобразовании типов данных. В этом случае выдается предупреждающее сообщение, и управление передается в начало цикла (ввод должен быть повторен).

Замечание: Если не выполнить инициализацию переменных перед инструкцией `input()`, то при возникновении исключительной ситуации (при вводе нажат только Enter), управление будет передано в модуль `except`, где возникнет новая исключительная ситуация в условном операторе `if`: переменная `a` неопределена.

Вывод данных

Вывод данных на экран монитора может быть выполнен функцией `print()`. Эта функция позволяет выполнять вывод как в Си-подобном формате, так и с использованием форматной строки.

Следующие примеры демонстрируют, как можно форматировать вывод.

```
for x in range(1,11):
    print('%2d %3d %7.2f' % (x, x*x, x*x*x))

print("{0:.2f} {1:.2f} {2:.4f}".format(a, x, y))
```

Буква в формате числа определяет тип выводимого числа. Так, `d` – это целый тип, `f` – вещественное число. Число в формате означает то число позиций, которое будет использовано для вывода числа. Для вещественного числа указывается, после точки, количество выводимых десятичных знаков.

В первом примере использован Си-подобный формат, в котором формат числа начинается с процента `"%"`, а во втором примере используется форматная строка, в которой формат числа задается в фигурных скобках.

Обратите внимание на то, что сами форматные строки начинаются и завершаются одиночной или двойной кавычкой. В Python допускаются оба вида кавычек для выделения строки. Важно только, чтобы начало и конец были одинаковыми.

Так же следует понимать, что в промежутках между символами форматирования могут находиться и другие символы или слова:

```
print('x=%2d x^2=%3d x^3=%7.2f' % (x, x*x, x*x*x))
print("a={0:.2f} x={1:.2f} y={2:.4f}".format(a, x, y))
```

Вернемся к нашим примерам и запишем их, используя правила языка Python:

```
1. y = tan(x**2/2-1)**2+(2*cos(x-pi/6))/(1/2+sin(a)**2)
```

Второе выражение представим в виде двух:

```
2. tmp = log(3-cos(pi/4+2*x), 3+sin(x))/(1+tan(2*x/pi)**2) y =
   pow(2, tmp)
```

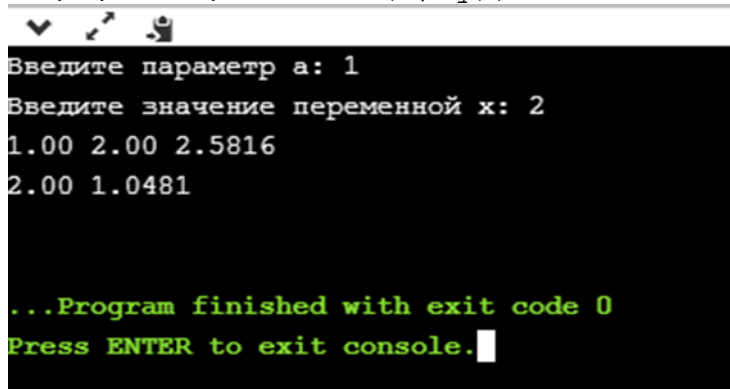
Описание алгоритма

Для вычислений необходимо обеспечить ввод двух переменных x и a . Поскольку по условиям задачи их тип и точность представления не заданы, выберем для них вещественный тип (*float*). Для оптимизации записи выражения используем промежуточную переменную *tmp*.

1. Ввести значения a и x , преобразовать к типу *float*.
2. Вычислить выражение 1.
3. Вывести результат вычисления.
4. Вычислить значение переменной *tmp*;
5. Вычислить выражение 2.
6. Вывести результат вычисления.

Листинг программы

```
# -*- coding: cp1251 -*- from
math import *
a = float(input('Введите параметр a: '))
x = float(input('Введите значение переменной x: '))
y=tan(x**2/2-1)**2+(2*cos(x-pi/6))/(1/2+sin(a)**2)
print("{0:.2f} {1:.2f} {2:.4f}".format(a, x, y))
tmp=log(3-cos(pi/4+2*x), 3+sin(x))/(1+tan(2*x/pi)**2)
y=pow(2, tmp)
print("{0:.2f} {1:.4f}".format(x, y))
```



```
Введите параметр a: 1
Введите значение переменной x: 2
1.00 2.00 2.5816
2.00 1.0481

...Program finished with exit code 0
Press ENTER to exit console.
```

Задание 1.1

Напишите программу для расчета по двум формулам. Подготовьте не менее пяти тестовых примеров. Предварительно выполните вычисления с использованием калькулятора или Excel (результаты вычисления по обеим формулам должны совпадать). используйте не менее пяти значений переменных

1. $z_1 = 2 \sin^2(3\pi - 2\alpha) \cdot \cos^2(5\pi + 2\alpha);$
 $z_2 = \frac{1}{4} - \frac{1}{4} \sin\left(\frac{5}{2}\pi - 8\alpha\right).$
2. $z_1 = \cos \alpha + \sin \alpha + \cos 3\alpha + \sin 3\alpha;$
 $z_2 = 2\sqrt{2} \cos \alpha \cdot \sin\left(\frac{\pi}{4} + 2\alpha\right).$
3. $z_1 = \frac{\sin 2\alpha + \sin 5\alpha - \sin 3\alpha}{\cos \alpha + 1 - 2 \sin^2 2\alpha};$ $z_2 = 2 \sin \alpha .$
4. $z_1 = \frac{2 \cdot \cos \alpha \cdot \sin 2\alpha - \sin \alpha}{\cos \alpha - 2 \cdot \sin \alpha \cdot \sin 2\alpha};$ $z_2 = \operatorname{tg} 3\alpha .$
5. $z_1 = 1 - \frac{1}{4} \sin^2 2\alpha + \cos 2\alpha;$ $z_2 = \cos^2 \alpha + \cos^4 \alpha .$
6. $z_1 = \cos \alpha + \cos 2\alpha + \cos 6\alpha + \cos 7\alpha;$
 $z_2 = 4 \cos \frac{\alpha}{2} \cdot \cos \frac{5}{2}\alpha \cdot \cos 4\alpha .$
7. $z_1 = \cos^2\left(\frac{3}{8}\pi - \frac{\alpha}{4}\right) - \cos^2\left(\frac{11}{8}\pi + \frac{\alpha}{4}\right);$ $z_2 = \frac{\sqrt{2}}{2} \sin \frac{\alpha}{2} .$
8. $z_1 = \cos^4 x + \sin^2 y + \frac{1}{4} \sin^2 2x - 1;$
 $z_2 = \sin(y+x) \cdot \sin(y-x) .$
9. $z_1 = (\cos \alpha - \cos \beta)^2 - (\sin \alpha - \sin \beta)^2;$
 $z_2 = -4 \sin^2 \frac{\alpha - \beta}{2} \cdot \cos(\alpha + \beta) .$
10. $z_1 = \frac{\sin\left(\frac{\pi}{2} + 3\alpha\right)}{1 - \sin(3\alpha - \pi)};$ $z_2 = \operatorname{ctg}\left(\frac{5}{4}\pi + \frac{3}{2}\alpha\right) .$
11. $z_1 = \frac{1 - 2 \sin^2 \alpha}{1 + \sin 2\alpha};$ $z_2 = \frac{1 - \operatorname{tg} \alpha}{1 + \operatorname{tg} \alpha} .$
12. $z_1 = \frac{\sin 4\alpha}{1 + \cos 4\alpha} \cdot \frac{\cos 2\alpha}{1 + \cos 2\alpha};$ $z_2 = \operatorname{ctg}\left(\frac{3}{2}\pi - \alpha\right) .$
13. $z_1 = \frac{\sin \alpha + \cos(2\beta - \alpha)}{\cos \alpha - \sin(2\beta - \alpha)};$ $z_2 = \frac{1 + \sin 2\beta}{\cos 2\beta} .$

14. $z_1 = \frac{\cos \alpha + \sin \alpha}{\cos \alpha - \sin \alpha}; \quad z_2 = \operatorname{tg} 2\alpha + \sec 2\alpha.$
15. $z_1 = \frac{\sqrt{2b+2\sqrt{b^2-4}}}{\sqrt{b^2-4+b+2}}; \quad z_2 = \frac{1}{\sqrt{b+2}}.$
16. $z_1 = \frac{x^2+2x-3+(x+1)\cdot\sqrt{x^2-9}}{x^2-2x-3+(x-1)\cdot\sqrt{x^2-9}}; \quad z_2 = \sqrt{\frac{x+3}{x-3}}.$
17. $z_1 = \frac{\sqrt{(3m+2)^2-24m}}{3\sqrt{m}-\frac{2}{\sqrt{m}}}; \quad z_2 = \sqrt{m}.$
18. $z_2 = \left(\frac{a+2}{\sqrt{2a}} - \frac{a}{\sqrt{2a+2}} + \frac{2}{a-\sqrt{2a}} \right) \cdot \frac{\sqrt{a}-\sqrt{2}}{a+2}; \quad z_2 = \frac{1}{\sqrt{a}+\sqrt{2}}.$
19. $z_1 = \left(\frac{1+a+a^2}{2a+a^2} + 2 - \frac{1-a+a^2}{2a-a^2} \right) \cdot (5-2a^2); \quad z_2 = \frac{4-a^2}{2}.$
20. $z_1 = \frac{(m-1)\sqrt{m} - (n-1)\sqrt{n}}{\sqrt{m^3n+nm+m^2-m}}; \quad z_2 = \frac{\sqrt{m}-\sqrt{n}}{m}.$
21. $z_1 = \frac{1+\sin^4(-a)-\cos^4(-a)}{\cos^2 a}; \quad z_2 = 2\operatorname{tg}^2 a.$
22. $z_1 = \sqrt{\frac{1-\sin(\frac{\pi}{2}-a)}{2}} + \sqrt{\frac{1+\cos(2\pi-a)}{2}}; \quad \pi < a < \frac{3\pi}{2}.$
 $z_2 = \sin \frac{a}{2} - \cos \frac{a}{2}; \quad \pi < a < \frac{3\pi}{2}.$
23. $z_1 = \left(\frac{1+6ac}{a^3-8c^3} - \frac{1}{a-2c} \right) : \left(\frac{1}{a^3-8c^3} - \frac{1}{a^2+2ac+4c^2} \right);$
 $z_2 = 1-2c+a.$
24. $z_1 = \frac{\frac{1}{a} - \frac{1}{b+c}}{\frac{1}{a} + \frac{1}{b+c}} \cdot \left(1 + \frac{b^2+c^2-a^2}{2bc} \right) : \frac{a-b-c}{abc}; \quad z_2 = \frac{a-b-c}{2} \cdot a.$
25. $z_1 = \frac{\sin^2(\pi+a) + \sin^2(\frac{\pi}{2}+a)}{\cos(\frac{3\pi}{2}+a)} \operatorname{ctg}(1.5\pi-a); \quad z_2 = \frac{1}{\cos a}.$
26. $z_1 = \frac{\operatorname{tg}(x-\frac{\pi}{2})\cos(\frac{3\pi}{2}+x) - \sin^3(3.5\pi-x)}{\cos(x-0.5\pi)\operatorname{tg}(1.5\pi+x)}; \quad z_2 = \sin^2 x.$
27. $z_1 = a^{\sqrt{\log_a b}} - b^{\sqrt{\log_b a}} + \operatorname{tg}(ab+3\pi/2); \quad z_2 = \operatorname{tg}(ab+\frac{3}{2}\pi);$

28. $z_1 = 2 \sin^2(3\pi - 2\alpha) \cdot \cos^2(5\pi + 2\alpha);$
 $z_2 = \frac{1}{4} - \frac{1}{4} \sin(\frac{5}{2}\pi - 8\alpha).$
29. $z_1 = \cos \alpha + \sin \alpha + \cos 3\alpha + \sin 3\alpha;$
 $z_2 = 2\sqrt{2} \cos \alpha \cdot \sin(\frac{\pi}{4} + 2\alpha).$
30. $z_1 = \frac{\sin 2\alpha + \sin 5\alpha - \sin 3\alpha}{\cos \alpha + 1 - 2 \sin^2 2\alpha}; \quad z_2 = 2 \sin \alpha.$

Таблица 1. Функции и методы в Python

Константы (Math)	Описание и пример использования
pi	Возвращает число π . <code>math.pi</code> => 3.141592653589793
e	Возвращает число e. <code>math.e</code> => 2.718281828459045
Функции (Sys)	
<code>int([<Объект>[,<Система счисления>]])</code>	Преобразует объект в целое число. Второй параметр указывает систему счисления <u>преобразуемого числа</u> . По умолчанию используется десятичная система счисления. <code>>>>int(7.5), int('71',10), int('0o71',8)⁵</code> (7, 71, 57) <code>>>>int("0xA",16)</code> 10 <code>>>>int(), int("0b11111111",2)</code> (0, 255)
<code>float([<Число или строка>])</code>	Преобразует целое число или строку в вещественное число. <code>>>>float(7), float("7.1"), float("12.")</code> (7.0, 7.1, 12.0) <code>>>> float("inf"), float("-inf"), float("nan")</code> (inf, inf, nan)
<code>bin(<Число>)</code>	Преобразует десятичное число в двоичное. Возвращает строковое представление числа. <code>>>>bin(255), bin(1), bin(-45)</code> ('0b11111111', '0b1', '-0b101101')
<code>oct(<Число>)</code>	Преобразует десятичное число в восьмеричное. Возвращает строковое представление числа. <code>>>>oct(7), oct(8), oct(64)</code> ('0o7', '0o10', '0o100')
<code>hex(<Число>)</code>	Преобразует десятичное число в шестнадцатеричное. Возвращает строковое представление числа. <code>>>>hex(10), hex(16), hex(255)</code> ('0xa', '0x10', '0xff')
<code>round(<Число>[, Кол-во знаков после точки])</code>	Вещественное число округляется до целого по следующему правилу: дробная часть меньше 0.5 – округление вниз; равна 0.5 – округление до четного числа; больше 0.5 – округление вверх. <code>>>>round(1.49), round(1.50), round(1.51)</code> (1, 2, 2) <code>>>>round(2.49), round(2.50), round(2.51)</code> (2, 2, 3) Второй параметр определяет число цифр после запятой. В этом случае округление выполняется по правилу: отбрасываемая часть меньше 0.5 – округление вниз; больше или равно 0.5 – округление вверх.

⁵ В Python можно использовать одиночные и двойные прямые кавычки. Важно, чтобы начальная и конечная кавычки были одинаковыми. При наборе программы в Word необходимо настроить соответствующие пункты автозамены: Файл → Параметры → Правписание → Параметры автозамены (для Office 2010)

	>>>round(2.449,1), round(2.450,1), round(2.451,1) (2.49, 2.5, 2.5)
abs(<Число>)	Возвращает абсолютное значение.
pow(<Число>, <Степень>, [,<Делитель>])	Возвращает число, возведенное в степень. Степень может быть вещественного типа. Если указан третий параметр, то возвращается остаток от деления результата на значение этого параметра. >>>pow(3, 3), pow(3, 3, 2), pow(3, 3.25) (27, 1, 35.533998349717294)
max(<Список чисел через запятую>)	Возвращается максимальное число из списка.
min(<Список чисел через запятую>)	Возвращается минимальное число из списка.
sum(<Последовательность> [, <Начальное значение>])	Возвращает сумму значений элементов последовательности (списка, кортежа) плюс начальное значение. >>>sum([1, 2, 3], 7), sum((1, 2, 3)) (13, 6)
divmod(x, y)	Возвращает кортеж из двух значений: целая часть и остаток от деления: $x // y$ и $x \% y$.
Функции (Math)	
sin(x), cos(x), tan(x)	Стандартные тригонометрические функции. Угол задается в <u>радианах</u> .
asin(x), acos(x), atan(x)	Обратные тригонометрические функции
degrees(x)	Преобразование радиан в градусы
radians(x)	Преобразование градусов в радианы
exp(x)	Экспонента
log(<Число>[, <База>])	Логарифм числа по основанию <База>. Если база не указана, то вычисляется натуральный логарифм.
log10(x), log2(x)	Десятичный и двоичный логарифмы (по базе 10 и 2 соответственно)
sqrt(x)	Квадратный корень
ceil(x)	Округление до ближайшего большего целого
floor(x)	Округление до ближайшего меньшего целого
fabs(x)	Возвращает абсолютное значение. тоже, что и abs(x)
fmod(x, y)	Остаток от деления. Тоже, что и $x \% y$.
factorial(n)	Факториал числа.
fsum(<список чисел>)	Возвращает точную сумму чисел из заданного списка. >>>sum([.1, .1, .1, .1, .1, .1, .1, .1, .1, .1]) 0.9999999999999999 >>>fsum([.1, .1, .1, .1, .1, .1, .1, .1, .1, .1]) 1.0