Лабораторная работа 6. Программирование.

Цель: знакомство с функциями операторов условия, цикла и процедурами.

# Синтаксис операторов.

* оператор условия имеет следующий синтаксис:

if булево\_выражение then последовательность\_операторов

[elif булево\_выражение then последовательность\_\_операторов]

[else последовательность\_операторов]

end if

в квадратных скобках [] – необязательные элементы.

* Операция if предназначена для использования в выражениях и аналогична тернарной операции условия (?) в языке С. Она имеет следующий синтаксис:

`if`(условие,операнд\_если\_да,операнд\_если\_нет);

* оператор цикла:

[for имя] [from выражение] [to выражение] [by выражение]

[while булево\_выражение]

do последовательность\_операторов end do;

Обязательным в данном случае является блок do…end do. Подобная конструкция определяет бесконечный цикл, прервать выполнение которого может только один из операторов: break, return, quit или возникновение ошибки при выполнении операторов цикла. Оператор next прекращает выполнение текущей итерации цикла и переключает оператор цикла на выполнение очередной итерации.

* Оператор цикла по элементам объекта

for имя in объект

[while булево\_выражение]

do последовательность\_операторов end do;

* Вывод сообщений из циклов условий или процедур можно реализовать функцией print или форматированный вывод – printf.

**Примеры использования**:

х:=5:

if x=0 then 0

elif x=l then 1

elif x=2 then 4/3

else print("Out of range!")

end if;



a:=4:b:=3:

c:=`if`(a>b, a,b)+sin(`if`(a>b,a,b));



Не выполняется ни разу:

for i from 1 to -1 do evalf(sqrt(i)) end do;

Выполняется два раза

for i from 1 to 2 do evalf(sqrt(i)) end do;





Определяет первое простое число, большее 500000, но меньшее 500010

for i from 5\*10^5 to 500010 while not isprime(i) do end do;

i;



> x:=2:

while x>1/2 do x:=x/2 end do;





Цикл по символам строки

for z in "ONE" do z end do;







Суммирование первых двух элементов списка

**M:=[x,y,z,t]: s:=0: j:=1:**

**for i in M do**

**s:=s+i;**

**if j>2 then break; end if;**

**j:=j+2;**

**end do:**

**s; j;**





Цикл только по нечетным значениям переменной цикла (пример использования next)

for i from 1 to 5 do

if is(i,even) then next end if; i; end do;







# Определение процедуры

Определение процедуры Maple имеет следующий общий синтаксис:

рrос([ список\_формальных\_параметров ])

[ local список\_\_локальных\_переменных; ]

[ global список\_глобальных\_переменных; ]

[ options список\_опций; ]

[ description строка\_описания; ]

последовательность\_операторов

end proc

При объявлении процедуры единственным обязательным параметром является последовательность операторов, формирующих тело процедуры. Остальные параметры, определяющие локальные и глобальные переменные, список формальных параметров, задающие специальные опции режима выполнения процедуры и строку описания, могут полностью отсутствовать.

Пример:

> vector\_length:=proc(x,y)

sqrt(x^2+y^2);

end proc;



После того, как процедура определена и ей присвоено имя, ее можно вызвать и выполнить с помощью оператора вызова функции, в котором задается имя процедуры, а в скобках определяются фактические переменные, которые заменяют в теле процедуры все формальные параметры:

> vector\_length(2,5);



Обычно возвращаемым значением процедуры является значение последнего вычисленного в теле процедуры оператора. В нашем примере последний, и единственный, оператор вычисляет квадратный корень из суммы квадратов параметров процедуры.

Если необходимо вернуть из процедуры не последний оператор, то необходимо пользоваться командой return.

В теле процедуры доступны системные переменные nargs и args. Значением первой из них является реальное количество фактических параметров при данном обращении к процедуре, а второй – последовательность фактических параметров при этом обращении.

Также можно использовать функции error и WARNING для вывода сообщений об ошибках и предупреждений.

error (mes, par1, par2, …)

WARNING(str, par1, par2, …)

## Задание 1. Простейшие циклы. По вариантам.

* 1. сделать циклом. В зависимости от задания проверить встроенной функцией вычисления произведения, суммы).
  2. Оформить сделанное в п. 1.1 в виде процедуры. (в зависимости от варианта в процедуру посылается eps или А или N или n) По номерам в журнале:

|  |  |
| --- | --- |
| Номер в журнале | Задание |
| 1, 14, 27 | Дана последовательность image002и ***eps>0***. Найти первый член последовательности image021для которого image022. |
| 2, 15, 28 | При ***a > 0*** задана последовательность image004. Найти при ***eps >0*** такое ***n***, что ***an < eps***. |
| 3, 16, 29 | Найти произведение image006Сомножители брать с точностью **eps>0**. |
| 4, 17, 30 | Найти сумму бесконечного рядаimage008 Суммировать до тех пор, пока члены ряда не станут меньше заданного ***eps>0***. |
| 5, 18, 31 | Найти сумму бесконечного ряда image010Суммировать до тех про, пока члены ряда не станут меньше заданного ***eps>0***. |
| 6 , 19, 32 | Среди чисел image012найти первое число, которое больше данного числа ***А>0***. |
| 7, 20, 33 | Дана последовательностьimage014. Найти сумму членов последовательности ***хn***, начиная с ***х0*** и до последнего ***хn***. |
| 8, 21, 34 | Найти сумму натуральных чисел, предшествующих заданному числу А. |
| 9, 22, 35 | Найти сумму image016, меньшую заданного числа ***А***. |
| 10, 23, 36 | Суммировать до тех пор, пока члены ряда не станут меньше ***eps>0***.  image018 |
| 11, 24, 37 | Найти сумму ряда image020. Суммировать до тех пор, пока члены ряда не станут меньше ***eps>0***. |
| 12, 25, 38 | Вывести все квадраты натуральных чисел, не превосходящие данного числа N. Пример: N=50 тогда 1 4 9 16... |
| 13, 26, 39 | Создать вектор случайных целых чисел размера N. Определить количество простых чисел в этом векторе. |

## Задание 2. Решить нелинейное уравнение методом деления пополам.

Дано произвольное уравнение в виде 

Для начала строим график функции  и определяем примерное положение корня – отрезок .

Для применения метода необходимо, чтобы выполнялось условие:



Если условие выполняется, то:

1. делим отрезок  пополам, середину обозначаем ;
2. проверяем условия:  
   если , то уменьшаем отрезок   
   если , то уменьшаем отрезок   
   если , то  ‑ искомый корень, выход.
3. Продолжаем действия 1) и 2) до тех пор, пока , где  ‑ заданная точность.

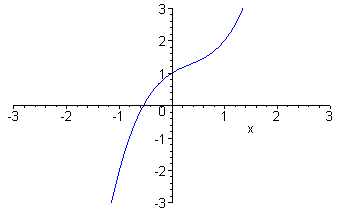
Строим график функции с найденным решением и всеми сделанными итерациями (построить все найденные ранее значения переменной ).

Пример:

Задана функция



Строим график



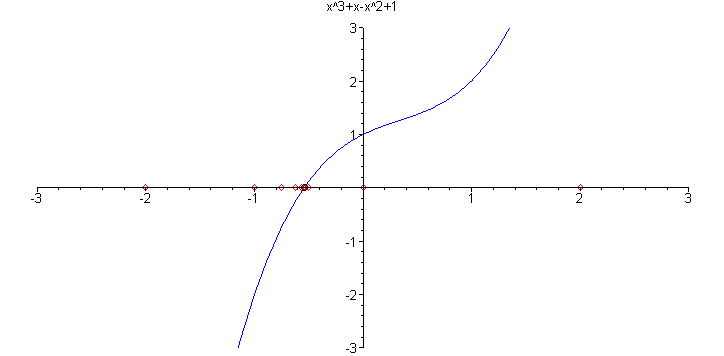
Выбираем отрезок : .

Задаем точность , вычисляем по данному алгоритму значение корня:



Корень был найден за 10 итераций.

График с итерациями:



## Задание 3. Написать процедуру для решения методом деления пополам.

Т.е. создать функцию dih:

> **dih(eq,a,b,e);**

Ее параметры:

**eq** – уравнение;

**a** – левый конец интервала;

**b** – правый конец интервала;

**e** – точность;

В результате выполнения этой процедуры должно выводиться:

Значение корня, количество итераций, точность, значение функции в найденной точке.  
> **dih(x^2-sin(x),0.5,2,0.1);**

x=0.968750, iterations=5 E=0.100000 f(x)=0.114298

А также должен строиться график:

