МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ БЕЛАРУСЬ

БЕЛОРУССКИЙ НАЦИОНАЛЬНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

***Факультет информационных технологий и робототехники***

Кафедра программного обеспечения вычислительной техники

и автоматизированных систем

**Отчет по лабораторной работе № 4**

по дисциплине: ”Функциональное и логическое программирование ”

на тему: ***”****Рекурсия****”***

Вариант g

Выполнил**:** студенты группы *10701118* Алейников Д.В.

Воробей И.А

Приняла**:** ст.пр. Ковалькова И.А.

Минск 2019

**Лабораторная работа № 4.** *Рекурсия***.**

**Цель работы:** приобретение практических навыков составления и откладки программ с использованием рекурсии.

***Задание 1:***

Возвести вещественное число a в целую степень n (степень n может быть положительной, нулевой или отрицательной). Составить два  варианта программы:

а) используя рекурсивное выражение  a^n=a^(n-1)\*a;

б) используя проверку четности степени: для нечетной степени используется рекурсивное выражение варианта а), для четной степени -  a^n = a^(n/2)\*a^(n/2).

**Код программы:**

**А)**

/\*

%Task 3 a)

predicates

nondeterm step(real,integer,real)

%step(a,n,r)(real,integer,real):(I,I,o)

nondeterm result

clauses

step(\_,0,1.0).

step(A,N,R) if N>0 and N1=N-1 and step(A,N1,R1) and R=R1\*A or N1=N+1 and

step(A,N1,R1) and R=R1/A.

result:-

write("Enter number A"),nl,

write("A="),readreal(A),

write("Enter extent N"),nl,

write("N="),readint(N),

step(A,N,R),

write(A,"^",N,"=",R),nl.

goal

result.\*/

**Б)**

/\*

%Task 3 b)

predicates

nondeterm step(real,integer,real)

%step(a,n,r)(real,integer,real):(I,I,o)

nondeterm result

clauses

step(\_,0,1.0).

step(A,N,R) if N>0 and N mod 2<>0 and N1=N-1 and step(A,N1,R1) and R=R1\*A.

step(A,N,R):- N<0,N mod 2<>0,N1=-N-1,step(A,N1,R1),R=1/(R1\*A).

step(A,N,R):- N>0,N mod 2=0,N1=N/2,step(A,N1,R1),R=R1\*R1.

step(A,N,R):- N<0,N mod 2=0,N1=-N-1,step(A,N1,R1),R=1/(R1\*A).

result:-

write("Введите вещественное число A"),nl,

write("A="),readreal(A),

write("Введите число N"),nl,

write("N="),readint(N),

step(A,N,R),

write(A,"^",N,"=",R),nl.

goal

result.\*/

***Задание 2:***

Используя рекурсию, запрограммировать решение следующих задач:

1. Найти сумму целых последовательных чисел в интервале от М до N (M<=N).

2. Просуммировать целые последовательные нечетные числа, т.е. 1+3+5+7+…, от 1 до N (учесть, что при вводе значения N оно может быть четным).

3. Вычислить значения следующих функций, используя разложение в ряд. Разработать два варианта программы – рекурсивный и итеративный, в итеративном варианте  задавать точность вычисления функции. Полученный результат сверить со значением соответствующей стандартной функции для вычисляемого аргумента.

а)  e = 1 + 1/1! + 1/2! + …+ 1/k! + … (k=0,1,2,…);

**Код программы:**

**1.**

/\*

%Task 4.1

predicates

nondeterm summ(integer,integer,integer)

%summ(n,m,r)(integer,integer,real):(I,I,o)

nondeterm result

clauses

summ(I,J,K) if I=J+1 and K=0.

summ(M,N,R) if M1=M+1 and summ(M1,N,R1) and R=R1+M.

result:-

write("Poke M"),nl,

write("M="),readreal(M),

write("Poke N"),nl,

write("N="),readint(N),

summ(M,N,R),

%R=R+M,

write("Summ from ",M," to ",N," = ",R),nl.

goal

result.\*/

**2.**

/\*

%Task 4.2

predicates

nondeterm summ(integer,integer,integer)

%summ(n,m,r)(integer,integer,real):(I,I,o)

nondeterm result

clauses

summ(F,N,R) if F=N+1 and R=0.

summ(F,N,R) if F1=F+1 and F mod 2<>0 and summ(F1,N,R1) and R=R1+F

or F1=F+1 and summ(F1,N,R1) and R=R1+0.

result:-

write("Poke N"),nl,

write("N="),readint(N),

summ(1,N,R),

write("Summ odds from 1 to ",N," = ",R),nl.

goal

result.\*/

**3.**

**a)**

%Task 4.3 iteration

/\*domains

n= integer

f=real

predicates

nondeterm factorial(integer,real)

%factorial(f,r)(integer,real):(I,o)

nondeterm factorial\_aux(integer,real,integer,real)

%factorial(f,r,f1,r1)(integer,real, integer,real):(I,o,I,o)

nondeterm step(real,integer,real)

%step(n,m,r)(integer,integer,real):(I,I,o)

nondeterm expo(integer,real,real)

%expo(n,m,r)(integer,real,real):(I,I,o)

nondeterm expo\_aux(integer,real,real,real,real)

%expo\_aux(n,m,n1,m1,r)( integer,real,real,real,real):(I,I,o,I,I)

nondeterm control(integer,real)

clauses

step(\_,0,1.0).

step(A,N,R) if N>0 and N mod 2<>0 and N1=N-1 and step(A,N1,R1) and R=R1\*A.

step(A,N,R):- N<0,N mod 2<>0,N1=-N-1,step(A,N1,R1),R=1/(R1\*A).

step(A,N,R):- N>0,N mod 2=0,N1=N/2,step(A,N1,R1),R=R1\*R1.

step(A,N,R):- N<0,N mod 2=0,N1=-N-1,step(A,N1,R1),R=1/(R1\*A).

factorial(N,F):- factorial\_aux(N,F,0,1).

factorial\_aux(N,F,L,P):-

L<N, NewL=L+1,NewP=P\*Newl,

factorial\_aux(N,F,NewL,NewP).

factorial\_aux(N,F,L,F):- L>=N.

expo(X,E,R):- expo\_aux(X,E,R,0,0).

expo\_aux(X,E,R,L,P):-

NewL=L+1,factorial(L,F),step(X,L,U),NewP=P+U/F,abs(P-NewP)>E,

expo\_aux(X,E,R,NewL,NewP).

expo\_aux(X,E,R,L,R):- L>=E,X>0.

control(X,G):-G=exp(X).

goal

write("Poke X="),readint(X),

write("Poke E(Aqurancy)="),readreal(E),

expo(X,E,R),

control(X,G),nl,

write("Standart function : e^",X,"=",G),nl,

write("e^",X,"=",R),nl.\*/

**b)**

/\*

%Task 4.3 recursion

domains

n= integer

f=real

predicates

nondeterm step(real,integer,real)

%step(n,m,r)(integer,integer,real):(I,I,o)

nondeterm factorial(n,f)

%factorial(f,r)(integer,real):(I,o)

nondeterm expo(integer,integer,real)

%expo(x,n,r)(real,integer,real):(I,I,o)

nondeterm result

clauses

step(\_,0,1.0).

step(A,N,R) if N>0 and N1=N-1 and step(A,N1,R1) and R=R1\*A or N1=N+1 and

step(A,N1,R1) and R=R1/A.

factorial(0,1).

factorial(N,F):- N>0,N1=N-1,factorial(N1,F1),F=F1\*N.

expo(\_,N,0) if N=100.

expo(X,N,R) if N1=N+1 and expo(X,N1,R1) and factorial(N,F) and step(X,N,U)

and R=R1+U/F.

result:-

write("Poke X"),nl,

write("X="),readint(X),

expo(X,0,R),

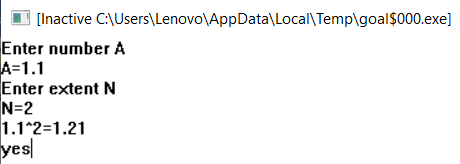
write("e^",X,"=",R),nl.

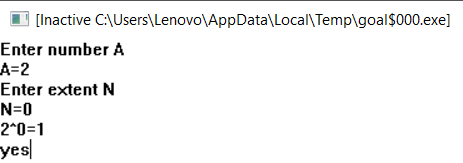
goal

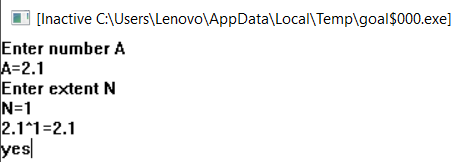
result.\*/

**Скриншоты:**

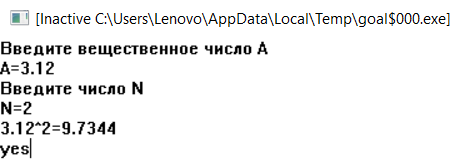
**А)**

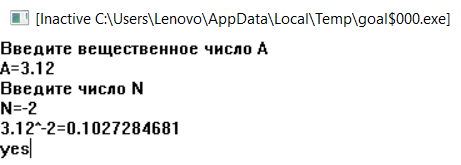




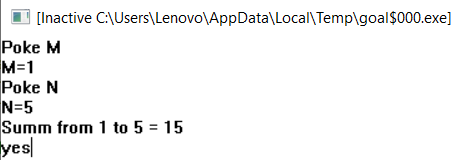


**Б)**

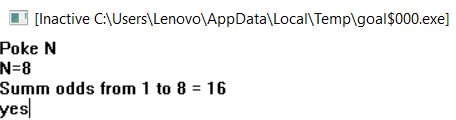


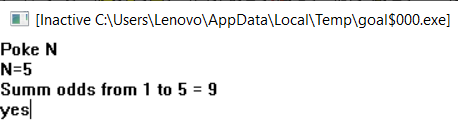


**1.**



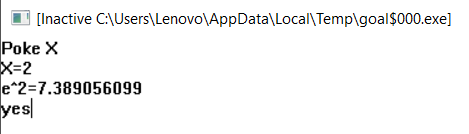
**2.**



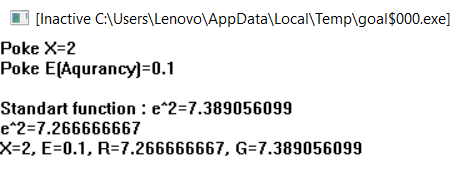


**3.**

**a)**



**b)**



**Контрольные вопросы:**

1. **Что такое рекурсия?**

Рекурсия - это определение  некоторого  отношения через  самого себя.

1. **Основная цель использования рекурсии в программах VisualProlog.**

В Прологе отсутствуют операторы цикла, то рекурсия служит основным средством программирования циклических процессов.

1. **Назначение граничного условия в рекурсивном правиле.**

 Для того чтобы рекурсивный метод решения был результативным, он должен в конце концов привести к задаче, решаемой непосредственно. Решить ее позволяют утверждения, называемые граничными  условиями.

1. **Принцип построения граничного условия.**

Граничное условие определяет параметры предиката при которых точно известен результат выполнения программы.

1. **Может ли быть в рекурсивном правиле более одного граничного условия?** Да.
2. **Как программируются разветвления в программах VisualProlog?**

При помощи рекурсии и итерации.

1. **Левосторонняя (нисходящая) и правосторонняя (восходящая) рекурсии.**

В процедуре вычисления факториала тело правила начинается с рекурсивного вызова определяемого предиката. Значит рекурсия называется левосторонней (нисходящей). Итеративный метод (восходящая рекурсия). Итеративной процедурой называется такая процедура, которая вызывает себя только один раз, причем этот вызов расположен в самом конце процедуры. Это так называемая правосторонняя рекурсия.

1. **Принципиальная разница между рекурсивной и итеративной процедурами.**

В процедуре вычисления факториала тело правила начинается с рекурсивного вызова определяемого предиката. Значит рекурсия называется левосторонней (нисходящей).Итеративный метод (восходящая рекурсия). Итеративной процедурой называется такая процедура, которая вызывает себя только один раз, причем этот вызов расположен в самом конце процедуры. Это так называемая правосторонняя рекурсия.

1. **Правила сопоставимости двух термов.**

Два терма сопоставимы, если они идентичны и переменные в обоих термах можно переписать в качестве значений объектов.

1. **Поиск с возвратом в программах VisualProlog.**

При поиске с возвратом цели должны быть доказаны по порядку. Для доказательства цели предложения просматриваются в том порядке в котором они записаны. Для того чтобы доказать головную цель необходимо доказать цели в теле правила.

1. **Поточный шаблон предиката.**

Известные параметры предиката называются входными (i), неизвестные – выходными (o). Совокупность входных и выходных параметров определяет работу предиката и называется **поточным шаблоном**. Например, предикат с двумя параметрами имеет 4 варианта поточного шаблона: (i,i), (i,o), (o,i), (o,o), где i – входной параметр, o – выходной параметр.

1. **Стандартные предикаты ввода-вывода VisualProlog.**

Write, writef, readint.

**Вывод:** мы приобрели основные практические навыки для составления и отладки программ с использованием рекурсии на VisualProlog.