Лабораторная работа 4. Символьные операции.

Цель: работа со списками, множествами и с уровнями вложенности выражения.

## Последовательности, списки, множества.

**Последовательность** – базовый тип данных, задает перечень объектов через запятую: s:=1,3,1,x,sin(y),0,-1/2; 

[s] – задает **список**: l:=[s]; 

{s} задает **множество**: m:={s}; 

**Список** – упорядоченный набор любых элементов.

**Множество** – неупорядоченный набор не повторяющихся элементов. Трактуется, как множество в математике.

**Выбор** элементов списка/множества осуществляется одинаково:

**l[1];**



**m[1];**



**l[2..4];** - со 2 по 4 элементы



**l[1..-1];** - выбрать все элементы



**Изменение** элементов списков/множеств

**l[1]:=5;l;** можно изменить





**> m[1]:=5;** нельзя изменить

Error, cannot reassign the entries in a set

**Создание** последовательностей функция seq или оператор $. Полный синтаксис в справке, несколько примеров:

****



****



**** третий параметр задает шаг



**a[i]$i=1..3;**



**x$10;**



**$1..10;**



## Работа с частями выражений

Функции для простейших манипуляций с выражениями:

 lhs – выделяет левую часть выражения;

 rhs – выделяет правую часть выражения;

 numer – выделяет числитель;

 denom – выделяет знаменатель.

## Работа с уровнями вложенности выражений

В общем случае выражения могут быть многоуровневыми и содержать объекты, расположенные на разных уровнях вложенности.

 nops(expr) – возвращает число объектов первого уровня (операндов) в выражении ехрr;

 ор(ехрr) – возвращает список объектов первого уровня в выражении ехрr;

 ор(n,ехрr) – возвращает n-й объект первого уровня в выражении ехрr.

Для списков/множетств возвращает **количество** **элементов**:

**nops([1,2,4,3]);**



## Преобразование выражений

Функции Maple работают с разными формами выражений и разными типами данных. Основной функцией для преобразования является функция convert:

 convert(expr, form, аrgЗ, ...)

Здесь ехрr - любое выражение, form - наименование формы (полный перечень в справке), аrg3, ... - необязательные дополнительные аргументы.

## Контроль за типами объектов

Выражения и их части в Maple рассматриваются как объекты. Функция whattype(object), возвращающая тип объекта, например string, integer, float, fraction, function и т. д.

## Функциональные преобразования элементов списков

Применение функции к операндам выражения:

map(fcn, expr, arg2, ..., argn)

Здесь fсn — процедура или имя, expr — любое выражение, argN — необязательные дополнительные аргументы для fсn.

Эта функция позволяет приложить fсn к операндам выражения ехрr:

**map(f,3+sin(x));**



**map(sin,[1,2,3]);**



**map(diff,[sin(x),x^2,3],x);**



# Задание 1. Генерация последовательностей.

* Задать последовательность **двумя** способами (через $ и через seq):



* Посчитать количество элементов в Х, обозначить n.
* Изменить элемент равный 0 на 1.
* Сгенерировать последовательность  **тремя** способами (через $, через seq и через map ):



* Сгенерировать вложенный список из Х и Y:



* Построить эти точки на графике.

# Задание 2. Уровни вложенности.

* Задать произвольное выражение f, содержащее не менее трех слагаемых/сомножителей. Например,   
   (взять **СВОЁ** выражение, на этом можно только проверить. Очень много вложений не надо, потом долго рисовать. Взять примерно как здесь).
* Функциями op, nops, whattype разложить его на составные части.   
  Например,  
  **n1:=nops(f);**в выражении 3 операнда

**f1:=op(f);** выделяем их

**whattype(f1[1]);** - смотрим тип первого объекта. 

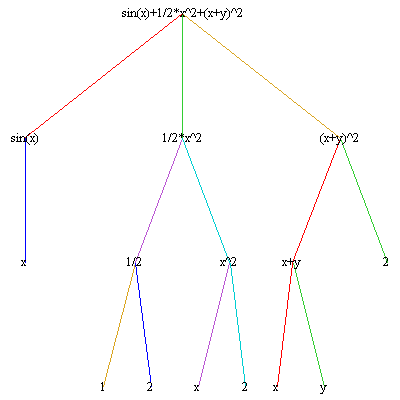




Здесь можно посмотреть сразу типы **всех** объектов, если использовать команду map. Ее результат (функция, умножение и степенная):



Далее разложить полученные составляющие до простейших объектов. Т.е. применить эти функции к каждому объекту несколько раз. Объекты переприсваивать переменным, ничего **не копировать**! В конце нарисовать структуру выражения в виде дерева. Для построения графика подключить пакет plots. Вывод текста – команда textplot (текст не набирать руками, а использовать команду convert), прямые строить обычным plot (указать две точки, он соединит их линией), объединить все вместе display.



# Задание 3.

* Найти корни 
* Обозначить (не копировать) их через X в виде списка.
* Найти его максимумы и минимумы, обозначить через переменные (выделить их можно с помощью op).
* Посчитать соответствующие значения функции в точках максимума и минимума.
* Построить график полинома и точек, построить текст, как на рисунке:

|  |  |
| --- | --- |
|  | Все переменные обозначаются заранее, как результат выполнения функции.  Для написания текста – функция textplot.  Для соединения текста и переменной испльзуется функция typeset:  Например, для заголовка графика (через f обозначена функция)  **title=typeset("График функции y = ",f);** |

# Задание 4.

Вычислить площадь фигуры, ограниченной параболами и . Для этого:

* Задать функции
* Найти их точки пересечения. Обозначить их a, b.
* Задать n=10 – число разбиений для построения штриховки.
* Посчитать длину интервалов штриховки 
* С помощью последовательности задать Х-координаты штриховки. (seq или $) (a+i\*h)



* Посчитать координаты функций f и g при данных Х. Использовать map (исходные функции должны быть заданы через ->). Обозначить их fY, gY.
* Для построения вертикальных линий сгенерировать последовательность координат начала и конца линии из списков: начало в X[i],fY[i], конец в X[i],gY[i].
* Вычислить площадь через интеграл.
* Построить графики функций, точек, линий, совместить все display

