Теоретические сведения к л.р. №3

Условный оператор

Одним из основных операторов реализующим ветвление в большинстве языков программирования является условный оператор `if`. Существует обычная и расширенная формы оператора `if` *Scilab*. в Обычный `if` имеет вид:

```
if условие
    операторы_1
else
    операторы_2
end
```

Здесь `условие` - логическое выражение; `операторы_1`, `операторы_2` - операторы языка *Scilab* или встроенные функции. Оператор `if` работает по следующем алгоритму:

- если условие истинно то выполняются , `операторы_1`
- если ложно - `операторы_2`

В *Scilab* для построения логических выражений могут использоваться условные *Scilab* операторы:

- `&` или `and` (логическое и)
- `|` или `or` (логическое или)
- `~` или `not` (логическое отрицание)
- операторы отношения:
  - `<` - меньше
  - `>` - больше
  - `>=` - больше или равно
  - `<=` - меньше или равно
  - `/==` - равно
  - `<>` - не равно

Зачастую при решении практических задач недостаточно выбора выполнения или невыполнения одного условия. В этом случае нужно воспользоваться расширенной формой оператора `if`:

```
if условие_1
    операторы_1
elseif условие_2
```

```
    операторы_2
elseif условие_3
    операторы_3
...
elseif условие_n
    операторы_n
else
    операторы
end
```

Пример:

```
a = 5;

if a > 0:
    disp(">");
elseif a < 0:
    disp("<");
else
    disp("=");
end
```

Цикл

Для создания циклов используется оператор `for`:

```
for x = xn:hx:xk
    операторы
end
```

Здесь `x` - имя скалярной переменной параметра цикла, `xn` - начальное значение параметра цикла, `xk` - конечное значение параметра цикла, `hx` - шаг цикла. Если шаг цикла равен 1, то `hx` можно опустить, и в этом случае оператор `for` будет таким:

```
for x = xn:xk
    операторы
end
```

Выполнение цикла начинается с присвоения параметру стартового значения `x = xn`. Затем следует проверка: не превосходит ли параметр конечное значение `x > xk`. Если `x > xk`, то цикл считается завершенным и управление передается следующему

за телом цикла оператору. Если же `x <= xk`, то выполняются операторы в теле цикла. Далее параметр цикла увеличивает свое значение на `hx` (`x = x + hx`). После чего снова производится проверка значения параметра цикла и алгоритм повторяется.

Пример:

```
vector = [0 1 2 3]
vector_size = size(vector)
vector_size = vector_size(2)

for i = 1:vector_size
    disp(vector(i))    // вывод элементов вектора
end
```

Функции

Функции - это специальные блоки кода, которые выполняют определенные задачи. Они помогают сделать программу более структурированной и позволяют многократно использовать один и тот же код в разных частях программы. Функция в *Scilab* определяется следующим образом:

```
function [output1, output2, ..., outputN] = functionName(input1, input2,
..., inputM)
    // Тело функции
    // Операции с входными параметрами для получения выходных значений
    output1 = ...;
    output2 = ...;
    ...
    outputN = ...;
endfunction
```

Здесь:

- `functionName` - имя функции
- `input1, input2, ..., inputM` - входные параметры функции
- `output1, output2, ..., outputN` - значения, которые возвращает функция
- тело функции содержит операторы для работы с входными данными и вычисления результата

Вызов функции выполняется следующим образом:

```
[output1, output2, ..., outputN] = functionName(nput1, input2, ..., inputM)
```

Пример:

```
function y = line_func(k, b, x)
    y = k * x + b;
endfunction

b = 3;
y = line_func(5, b, 10)
```

Вычисление точек пересечения

Для решения задачи необходимо:

- найти коэффициенты уравнения прямой, которая пересекает фигуру
- найти точки пересечения этой прямой с прямыми, заданными сторонами фигуры, решая соответствующие системы уравнений.

Для построения линейной функции исользуется следующая формула:

$$y = kx + b$$

Пусть задан отрезок AB с точками $A[x_1, y_1]$ и $B[x_2, y_2]$. Тогда коэффициенты уравнения линейной функции можно вычислить следующим способом:

$$k = \frac{y_2 - y_1}{x_2 - x_1}$$

$$b = \frac{x_2 y_1 - y_2 x_1}{x_2 - x_1}$$

Для нахождения точки пересечения двух прямых необходимо решить систему линейных уравнений вида:

$$\begin{cases} k_{f1} x_{f1} + y_{f1} + b_{f1} = 0 \\ k_{f2} x_{f2} + y_{f2} + b_{f2} = 0 \end{cases}$$

Решение этой системы уравнений в общем виде:

$$x_c = \frac{b_{f2} - b_{f1}}{k_{f1} - k_{f2}}$$

$$y_c = k_{f1} \frac{b_{f2} - b_{f1}}{k_{f1} - k_{f2}} + b_{f1}$$

Эти формулы применимы только тогда, когда прямые не параллельны осям ординат (т.е. не горизонтальны и не вертикальны)

Если одна из прямых параллельна оси Ox (т.е. координата y_{f1} у всех точек одинакова и $y_{f1} = C$), то для нахождения координат точки пересечения используются следующие

формулы:

$$x_c = \frac{C - b_{f2}}{k_{f2}}$$

$$y_c = C$$

Если одна из прямых параллельна оси Oy (т.е. координата x_{f1} у всех точек одинакова и $x_{f1} = C$), то для нахождения координат точки пересечения используются следующие формулы:

$$x_c = C$$

$$y_c = k_{f2}C + b_{f2}$$

После нахождения точки пересечения прямых необходимо проверить лежит ли она на отрезке, представляющем собой сторону фигуры (AB с точками $A[x_1, y_1]$ и $B[x_2, y_2]$):

$$min(x_1, x_2) <= x_c <= max(x_1, x_2)$$

$$min(y_1, y_2) <= y_c <= max(y_1, y_2)$$

Алгоритм нахождения точек пересечения:

- вычисленяем коэффициенты b и k для пересекающей линии
- в цикле проходим по каждому отрезку фигуры
  - если отрезок фигуры горизонтальный, то вычисляем координаты точки пересечения используя соответствующую формулу
  - иначе если отрезок фигуры вертикальный, то вычисляем координаты точки пересечения используя соответствующую формулу
  - в остальных случаях вычисляем координаты точки пересечения используя стандартную формулу
  - выполняем проверку: лежит ли найденная точка на отрезке фигуры и выводим соответствующее сообщение и координаты точки фигуры

Подсказка:
Реализуйте следующие функции:

- для вычисления коэффициента *k*
- для вычисления коэффициента *b*
- для вычисления точки пересечения в случае, если одна из прямых вертикальная
- для вычисления точки пересечения в случае, если одна из прямых горизонтальная
- для вычисления точки пересечения в стандартном случае
- для проверки лежит ли точка пересечения на отрезке стороны фигуры

Реализовать цикл по каждой стороне фигуры можно следующим образом:

```
// матрица точек 2D фигуры (условная)
fig = [0 0; 1 0; 0 0; ...]

// вычисляем размер матрицы
fig_size = size(fig)
// количество точек (количество строк в матрице)
points_count = fig_size(1)

// цикл
for i = 1:(points_count - 1):
    // отрезок фигуры (разберитесь сами как он был получен
    // и что он из себя представляет)
    fig_line = fig(i:i + 1, :)

    // остальные операции

    // на первом этапе координаты точки пересечения
    // можно просто выводить на консоль после каждой
    // итерации цикла
end
```