創好リナの

Programming Seminar

5bit

Function

2018.09.09

関数

- 前回の問題解説
- 関数とは
- Luaの関数
- Luaの標準関数
- 前回の問題を関数を使って

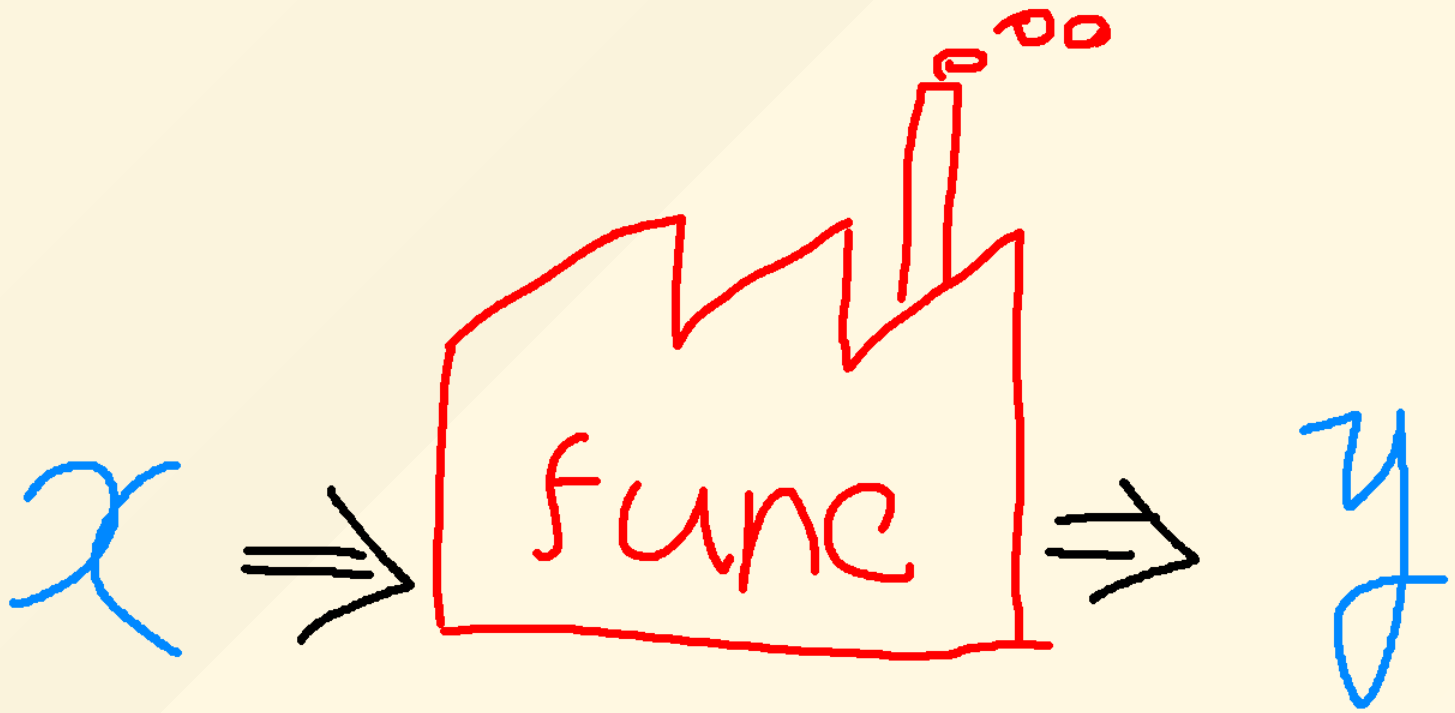
前回の4-3解説

関数を使って
わかりやすく書こう

関数 #とは

関数 Function

- 値を入力 -> 値が出力される処理の1かたまり
- プログラミングでは動作のみを表す物もある



関数を作るタイミング

- 関数を作る基本は「**同じような処理を一つにする**」
- 処理の意味ごとにも分ける事が多い
  - 初期化処理など
- 1関数は1画面に収まる程度が最適
  - 多くて40行程度
-

いろいろな関数

- 再帰関数
  - 関数内で自分自身を呼び出しループのような処理をする関数
  - 簡潔に書ける場合があるが慣れてないと死ぬ
- メソッド
  - オブジェクト指向とかでよく使う
  - クラス等の中に含まれる関数

Luaの関数

関数の作り方

- `function 関数名(引数名リスト)` ~ `end`
- 値を返す時は`return 式`
  - `return`を使うと関数の途中でも関数を抜ける!
- 使う時は`関数名(実引数リスト)`
  - 関数実行は値を返す(ない場合も)ので`式`

```
function add(a, b)
    local ret = a + b
    return ret
end

print(add(3, 5)) -- 8
```

Luaの関数Tips

- `function(引数名リスト)` ~ `end`で
  function型を返す式になる!
  - 実は`関数名 = function()`に変換されてる
    -> 関数のグローバル変数ができてる
- `local function 関数名(引数名リスト)`でローカル関数が
  できる
- `変数:関数名(引数...)`で`関数名(変数, 引数...)`
  - メタテーブルという物を使う時
  - Luaのメソッド呼び出し
  - 詳しくは今度!

main関数のすゝめ

- Luaはmain関数を作らずそのまま書いても動くが、main関数を作ることでスコープを汚さずに書ける

```
function main()
  ...
end

main()

-- 引数使用
function main(args)
  ...
end

main({...})
```

Luaの標準関数

- Lua 5.1 リファレンスマニュアル
  - http://milkpot.sakura.ne.jp/lua/lua51_manual_ja.html
  - Lua5.1で検索

よく使う

- table.*
  - テーブル操作(配列とか)は基本これ使う
  - `concat`/`insert`/`remove`/`sort`/`maxn`
- string.*
  - 文字列操作
  - 実は`文字列型変数:関数名()`で使える!

よく使う

- math.*
  - 数学的なやつ全部
  - `max`/`min`/`abs`/三角関数/対数 etc...
- io.*
  - ファイル操作系
  - 今後確実に使う

ComputerCraftの関数

- ComputerCraft Wiki (公式)
  - http://www.computercraft.info/wiki/Main_Page
  - 英語だけど最新情報でわかりやすい
- ComputerCraft 非公式JapanWiki
  - 日本語だけど情報が古い&全部は書かれていない
  - 基本的なものは記載されている

4-3を関数を使って書いてみよう

関数はプログラミングを
やっていれば100%使う

以上

5bit目 問題

5-1

- 入力された数配列の平均値を返す関数を作る
- 標準入力で負の値が入力されるまで数値を入力し、平均を出すプログラムを作りなさい
  - 負の値は含まない

```
num1: 100[enter]
num2: 80[enter]
num3: 60[enter]
num3: 75[enter]
num4: 89[enter]
num5: 64[enter]
num6: -1[enter]

Average: 78.0
```

5-2

- 3ブロック間隔で2*2に植えられた白樺を伐採するプログラムを作りなさい
- スタート地点の右に苗木チェスト、左に木のチェストがあるものとする

5-3

- 横1＊縦3で指定した数掘り進むプログラムを作りなさい
- ただし、砂利や砂などの落下を考慮し、全て掘る事

```
length: 10[enter]
```