

システムソフトウェア特論#1

学籍番号：1931150 名前：柳澤 士朗

2020/2/27

1 演習 1-1

次の計算を C 言語の「関数型」スタイルで書いてみよ。

a N を与えて $\sum_{i=1}^N i$

- 方針: $n = 1$ ではない場合において $n + f(n - 1)$ を計算するようにプログラムを設計する。
- ソースコード

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 int f(int);
5
6 int main(int argc, char *argv[]){
7     printf("%d\n", f(atoi(argv[1])));
8 }
9
10 int f(int n){
11     if(n == 1) {return 1;}
12     else      {return n + f(n-1);}
13 }
```

- 実行結果

```
adminnoMacBook-ea:Program Jiro$ ./ex1-1 10
55
```

- 考察

与えられた値に対して正しい解が出力されていることから, このプログラムは正しいと考えられる.

b N を与えて $\sum_{i=1}^N i^2$

- 方針: a と同様に $n = 1$ ではない場合において $n \times n + f(n - 1)$ を計算するようにプログラムを設計する。
- ソースコード

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 int f(int);
5
6 int main(int argc, char *argv[]){
7     printf("%d\n", f(atoi(argv[1])));
8 }
9
10 int f(int n){
11     if(n == 1) {return 1;}
12     else      {return n * n + f(n-1);}
13 }
```

- 実行結果

```
adminnoMacBook-ea:Program Jiro$ ./ex1-1 3
14
adminnoMacBook-ea:Program Jiro$ ./ex1-1 4
30
adminnoMacBook-ea:Program Jiro$ ./ex1-1 5
55
adminnoMacBook-ea:Program Jiro$
```

- 考察

与えられた値に対して正しい解が出力されていることから、このプログラムは正しいと考えられる。

c N を与えて $fib(N)$ ただし $fib(0) = fib(1) = 1, fib(n) = fib(n-1) + fib(n-2) (n \leq 2)$

- 方針: $n = 0, n = 1$ の場合は 1 を返し, $n \leq 2$ の場合は $fib(n-1) + fib(n-2)$ を計算するようにプログラムを設計する。

- ソースコード

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 int f(int);
5
6 int main(int argc, char *argv[]){
7     printf("%d\n", f(atoi(argv[1])));
8 }
9
10 int f(int n){
11     if(n == 0 || n == 1) {return 1;}
12     else {return f(n-1) + f(n-2);}
13 }
```

- 実行結果

```
adminnoMacBook-ea:Program Jiro$ ./ex1-1 5
8
adminnoMacBook-ea:Program Jiro$ ./ex1-1 6
13
adminnoMacBook-ea:Program Jiro$ ./ex1-1 7
21
adminnoMacBook-ea:Program Jiro$ ./ex1-1 8
34
adminnoMacBook-ea:Program Jiro$
```

- 考察

与えられた値に対して正しいフィボナッチ数が出力されていることから、このプログラムは正しいと考えられる。

d その他の「関数型」スタイルで書くのに良いと思う任意の計算

- 今回は N を与えて $\prod_{i=1}^N i^2$ を求めるアルゴリズムを設計する。方針は a, b と同様に考え, $i = 1$ ではない場合において $n \times n \times f(n-1)$ を計算するようにアルゴリズムを設計した。

- ソースコード

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 int f(int);
5
6 int main(int argc, char *argv[]){
7     printf("%d\n",f(atoi(argv[1])));
8 }
9
10 int f(int n){
11     if(n == 1) {return 1;}
12     else      {return n * n * f(n-1);}
13 }
```

- 実行結果

```
adminnoMacBook-ea:Program Jiro$ ./ex1-1 4
576
adminnoMacBook-ea:Program Jiro$ ./ex1-1 5
14400
adminnoMacBook-ea:Program Jiro$ ./ex1-1 6
518400
adminnoMacBook-ea:Program Jiro$
```

- 考察

与えられた値に対して正しい解が出力されていることから、このプログラムは正しいと考えられる。

2 アンケートの回答

Q1 プログラミングは好き/得意ですか、苦手ですか。それはなぜですか。どういうところが特にそう思う？
プログラミングは苦手である。なぜなら思うように実行できないときや、良い解を得られないときにどの部分でエラーを起こしているのか、プログラムの書き方が最適ではないかを探す作業がとても大変だからである。私自身現在研究でプログラミングを行っているが、上手く実行できないことが度々起こるのでそのような点からもプログラミングに苦手意識を持ってしまっていると考えられる。

Q2 様々なプログラミング言語があるということについてどのように考えていますか。
一言にプログラミング言語といっても、人によってプログラミング言語を用いて実現したいことは異なると考えられる。実現のためにそれに適したプログラミング言語が出現するのは最もな考え方だと思う。また似たような分野に特化したプログラミング言語でも人によって好き嫌いや、得意不得意な言語というのはあるだろうから、様々なプログラミング言語があることについては仕方がないことであるし、合理的であるとも考える。

Q3 リフレクション、感想。
関数型スタイルで書くことによってシンプルに 2,3 行で求めたい計算の実装ができてしまう。確かにループでも同じ様に計算して求めることは可能であると思うが、関数型スタイルの方が煩雑になる可能性は低いと考えられる。そのような点から関数型スタイルで実装できるものについてはループよりも関数型スタイルで実装した方が良いと今回の課題を通して実感した。