てけとーにfizzbuzzやりました

2017年12月16日

姫路IT系勉強会 2017.12

(一部加筆訂正)

さとう

とりあえず３つ

- 普通のfizzbuzz
  - If {} else if {} else…
- 馬鹿馬鹿しいfizzbuzz
  - printf(“fizzbuzz\n”);
- 少し真面目なfizzbuzz
  - car(cdr(cdr(cdr list)))

まずは普通のfizzbuzzです

```
$ echo $NUMBER | dc -f loop.txt
1
2
fizz
4
buzz
...
fizzbuzz
```

dc(1)について - 1

- man 1 dc
- スタック(先入れ後出し)とレジスタ(データの置き場所)を備えた計算器です
- 数値や文字列をスタックに積み上げたり、レジスタに出し入れしたり、スタックの値を加減乗除したりできます。
- 単行演算子ならスタックの最後を加工
- 二項演算子ならスタックの頂上２つ
- 実はレジスタも「スタック」が使える(まだ使えてないです)

dc(1)について - 2

- 文字列は、[hogeほげ%1234]などと書きます。
- 文字列の中身を、「コマンド」として実行できます(マクロ)

```
[ 10 10 * p] x
=> 100
```

さきほどのfizzbuzz、実は...

- **$ echo 16 | dc -e "0s01s13s35s515sz[lqlz%stltl0=u]se[lql5%stltl0=k]sb[lql3%stltl0=i]sf[l00=p]sc[lqp]sp[[fizzbuzz]p_1s0]su[[buzz]p_1s0]sk[[fizz]p_1s0]si[0s0]so[lexlbxlfxlcxloxl9x]s60s7?1+s8[l71+sqlqs7cl7l8!=6]s9l9x"**

  1

  2

  fizz

  ...

難読ごめんなさい

- レジスタ名が「英数字一文字」しか使えない
- 構文のほとんどで、スペース、改行を無視する
- そのため、以下がfizzbuzzとして機能します。

"
0s01s13s35s515sz[lqlz%stltl0=u]se[lql5%stltl0=k]sb[lql3%stltl0=i]sf[l00=p]sc[lqp]sp[[fizzbuzz]p_1s0]su[[buzz]p_1s0]sk[[fizz]p_1s0]si[0s0]so[lexlbxlfxlcxloxl9x]s60s7?1+s8[l71+sqlqs7cl7l8!=6]s9l9x
"

多少マシに書けば

- こんな風になると思います。

  0 s0

  1 s1

  3 s3

  5 s5

  [[fizzbuzz] p _1 s0] su

  [[buzz] p _1 s0] sk

  [[fizz] p _1 s0] si

  (以下略)

次に、馬鹿馬鹿しいのいきます

```
$ ruby mkfz_c.rb $NUMBER
#include<stdio.h>
int main () {
        printf("1\n");
        printf("2\n");
        printf("fizz\n");
.
.
.
        print($NUMBER\n);
  return 0;
}
```

蛇足ながら解説

- スクリプト言語から、「計算しない」C言語のソースを吐かせてみました。
- ビルドすればそれでfizzbuzzになるはずです。
- あまりにも馬鹿馬鹿しいので、数が大きいとビルドできなかったり、あるいはもっと変なことになったりする、かもしれません

  (以前、ビルド中にOSが反応しなくなりました...)

最後に、少し真面目にします

- fizzbuzzは、

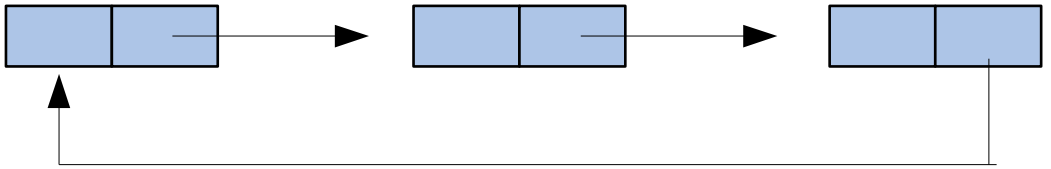
  1 2 fizz 4 buzz fizz 7 8 fizz buzz 11 fizz 13 14 fizzbuzz
  16 17 fizz 19 buzz fizz 22 23 fizz buzz 26 fizz 28 29 fizzbuzz
  31 32 fizz 34 buzz fizz 37 38 fizz buzz 41 fizz 43 44 fizzbuzz
  46 47 fizz 49 buzz fizz 52 53 fizz buzz 56 fizz 58 59 fizzbuzz

- 1から”fizzbuzz”まで15とおり
- 16以上は、同じパターンの繰り返し
- ということは、わざわざ毎回割り算しなくても
- 配列を使ってみました。

(片方向の)循環リスト

- 最初の次が最後

  要するに輪っかになってると。



- 普通のリストなら、最後の次はないけど
- こちらはいくらでも「次」を手繰れると

LISPではありふれてるようで

- たとえばCLISPなら

  [1]>> (setf list '#1=(1 2 3 . #1#))

  #1=(1 2 3 . #1#)

  [2]>> (cdr list)

  #1=(2 3 1 . #1#)

  [3]> (cdr (cdr list))

  #1=(3 1 2 . #1#)

  [4]> (cdr (cdr (cdr list)))

  #1=(1 2 3 . #1#)

今回はCommonLispでなく

- Scheme系のgaucheつかいました。

  (まあどれもあまり違わないと思います...)

で、こんなのです。

- 循環リスト

  (define fzlist

    '#0=(#f #f "fizz" #f "buzz" "fizz" #f #f "fizz" "buzz"

          #f "fizz" #f #f "fizzbuzz" . #0#))

- 必要なだけ(cdr fzlist)します
- 最後に(car list)すればいい。

  文字列ならそれを返す

  偽なら数字を

あとはコードだけ

- すみません。長いので
- こっち見てください

- LISPな人にフォロー頂きました。
- ありがとうございます

今朝浮かんだこと

- ループ処理の代わりに、乱数でも使うか
- 一発で処理は完了しないけど、「いずれは」終わるに違いない...

- もう少し変なのが浮かぶといいのですが。

いつもどおり、薄っぺらいけど

- ご清聴ありがとうございます。

終