

便利な方法

- TABキーで変数名・関数名等の自動補完
- 実行した履歴（ヒストリ）を利用する
 - カーソルキーの上下
- 「Markdown」に対応したエディタ
 - Atom
- Terminalで邪魔に感じたら（クリアしたいなら）
 - Ctrl + L
- 探し方のヒント
 - 教科書の索引(index)を使う
 - その前にあるクイックリファレンスを使う
 - 自前で勉強中にオリジナルの索引をつくる。
- エディタ毎のコピペ機能を使えるようにしよう。
- プログラミングをするときのエディタは、インデントに対応してるものが望ましい。（Emacs, vim, ,,,）
- 過程（取り組み方）。どう課題に取り組もうとしたのか、を書きながら。その結果が想定通りだったのか。想定外だったのか。といったことを、レポートに残しながら、進めると。課題やレポート作成の時間が減るんじゃないかな。取り組みやすくなるんじゃないかな。

「第1引数が3だけで割り切れるなら'Fizz'を返す。」

- まずは「第1引数」をarg1に保存する。
- arg1が3で割り切れるかどうかを判断する。
- 割り切れるときに'Fizz'を返す。

```
>>> arg1 % 3
0
>>> arg1
```

```
3
>>> if arg1 % 3 == 0:
...     print('Fizz')
...
Fizz
```

「第1引数が5だけで割り切れるなら'Buzz'を返す。」

- さっきとほぼ一緒。5に置き換えてみる。

```
>>> if arg1 % 5 == 0:
...     print('Buzz')
...
Buzz
>>> arg1
10
```

```
... ]
    File "<stdin>", line 3
        ]
        ^
SyntaxError: invalid syntax
```

- 間違って[を入力したら上記のようなエラーが出た。

「第1引数が3と5の両方で割り切れるなら'FizzBuzz'を返す。」

- 条件が2つに増えた。
- 「and」

```
>>> arg1
10
>>> if (arg1 % 5 == 0) and (arg1 % 3 == 0):
```

```
...     print('FizzBuzz')
...
>>> arg1 = 15
>>> if (arg1 % 5 == 0) and (arg1 % 3 == 0):
...     print('FizzBuzz')
...
FizzBuzz
```

「第1引数が上記のどれにも当てはまらないなら、その数値をstr型に変換したオブジェクトを返す。」

- 「それ以外」「どれにも当てはまらない」というのが良くわからないので、まずはprintで確認してみる。
- 何度も同じコードを手打ちするのは面倒なので、ここからファイルに書くことにする。

```
# コード例
arg1 = 1
if arg1 % 3 == 0:
    print('Fizz')

if arg1 % 5 == 0:
    print('Buzz')

if (arg1 % 3 == 0) and (arg1 % 5 == 0):
    print('FizzBuzz')

print('それ以外')
```

```
#実行例
oct:tnal% python3 week4.py
それ以外
```

[~]

失敗例その1

```
def judge_number(arg1):
    arg1 = 1
    if arg1 % 3 == 0:
        print('Fizz')

    if arg1 % 5 == 0:
        print('Buzz')

    if (arg1 % 3 == 0) and (arg1 % 5 == 0):
        print('FizzBuzz')

    print(str(arg1))

judge_number(1)
judge_number(2)
```

- 敗因（原因分析）：関数化したつもりだが、引数であるarg1を関数内で上書きしてしまっていたため、常に同じ結果がしか返してくれない関数になってしまっていた。

```
def judge_number(arg1):
    print('今の数字は', arg1, 'だよ: ', end='')
    if (arg1 % 3 == 0) and (arg1 % 5 == 0):
        return 'FizzBuzz'

    if arg1 % 3 == 0:
        return 'Fizz'

    if arg1 % 5 == 0:
        return 'Buzz'

    return(str(arg1))

result = judge_number(1)
print('1の結果は', result, 'だよ')
result = judge_number(2)
print('2の結果は', result, 'だよ')
result = judge_number(3)
print('3の結果は', result, 'だよ')
result = judge_number(4)
print('4の結果は', result, 'だよ')
result = judge_number(5)
print('5の結果は', result, 'だよ')
result = judge_number(6)
```

```
print('6の結果は', result, 'だよ')
result = judge_number(7)
print('7の結果は', result, 'だよ')
result = judge_number(8)
print('8の結果は', result, 'だよ')
result = judge_number(9)
print('9の結果は', result, 'だよ')
result = judge_number(10)
print('10の結果は', result, 'だよ')
result = judge_number(11)
print('11の結果は', result, 'だよ')
result = judge_number(12)
print('12の結果は', result, 'だよ')
result = judge_number(13)
print('13の結果は', result, 'だよ')
result = judge_number(14)
print('14の結果は', result, 'だよ')
result = judge_number(15)
print('15の結果は', result, 'だよ')
```