第 3 回 python 講座資料

2018 年 7 月 18 日

1　例外処理

1.1　基本

python での例外処理は tryexcept 構文を用います。こちらの構文は以下の通りになります。

tryexcept

```
try:
    処理
except:
    エラー時の処理
```

例えば BMI を測定するプログラムを作成したとします。ここで身長と体重を入力してもらう必要があるのですが間違って 0 を入力してしまった場合 $ZeroDivisionError$ という 0 が入力されたというエラー、そもそもなにも入力されなかった場合は $ValueError$ というようなエラーが発生してしまいます。そこで間違って入力された場合例外処理によってもう一度入力されるようにプログラムを書く必要があります。そこで次のページプログラムを実行してみてください。これはＢＭＩを求めるプログラムです。

例

```
while true
   try:
      weight=float(input(”体重–>”))
      height=fliat(input(”身長–>”))
      height=height/100
      bmi=weight/(height*height)
      break
   except:
      print(”入力ミス、再度入力”)
result=””
if bmi  <  18.5:
   result=”痩せ型”
elif bmi <  25:
   result=”標準体重”
elif bmi < 30:
   result=”肥満 (軽)”
else:
   result=”肥満 (重)”
print(”BMI:”,bmi)
print(”判定:”,result)
```

次に tryexcept 文ではエラーによって処理を変えることことができます。これは except 文の後にエラー種類を記述することによってできます。では次のプログラムを実行してみてください。

エラーよって処理を変える

```
s=input("数字を入力:")
try:
   v=100/float(s)
   print(v)
except ValueError as e:
   print(e)
except ZeroDivisionError as e:
   print(e)
except:
   print("その他のエラー")
```

またエラーの発生に関わらず必ず行い操作は存在すると思います。例えばファイル操作を行う場合はエラーの発生に関わらずファイルを閉じる操作を行わないといけません。そこで使われる構文が finally です。これは tryexcept 文の最後に記述することで必ず行う処理を作ることができます。

finally

```
try:
   処理
except:
   エラー処理
finally:
   必ず行われる処理
```

2　ファイル操作

2.1　基本操作

python におけるファイル操作は主に 4 つの構文で構成されます。ファイルを開く open(), ファイルを読む read(), ファイルに書き出す write(), ファイルを閉じる close() がそれにあたります。

それでは簡単なプログラムを実行してみましょう。ここでは適当なファイル test.txt を用意してそのファイルに対して操作を行いましょう。次のプログラムを実行してみてください。

test.txt

```
keep on asking, and it will be given you:
keep on seeking, and you will find:
keep on knocking, and it will be opened to you:
```

ファイル操作の基本

```
a_file=open(”test.txt”,encoding=”utf-8”)
s=a_file.read()
a_file.close()
print(s)
```

コマンドライン

```
Keep on asking, and it will be given you:
Keep on seeking, and you will find:
Keep on knocking, and it will be opened to you:
```

このように a_file という形でファイルを開き、変数 s に読み込んだ結果を保存。その後 print(s) で表示しています。他の言語でも同じですがファイルは開いたら必ず閉じることを忘れないでください。

また今回は文字コードが $UTF-8$ だったためプログラム上で $UTF-8$ を宣言していますが日本語 ($Shift\_JIS$) を使いたい場合は以下のようにプログラミングします。

日本語の場合

```
a_file=open(”test.txt”,encoding=”sjis”)
```

次に書き込みを行いましょう。書き込む場合は先ほどのプログラムに $mode = ”w”$ を加え、read() を write() に変えることで実行できます。それでは次のプログラムを実行してみてください。

ファイル書き込み

```
a_file=open(”test2.txt”,mode=”w”,encoding=”utf-8”)
a_file.write(”好きな言葉を入力してください”)
a_file.close()
```

すると test2.txt というファイルが生成され書き込みが行われました。

このようにして書き込みを行います。open() には他にも mode が存在します。以下にどのようなものがあるかまとめた表を記します。

mode の値	解説
w	ファイルを書き込みモードで開く
r	ファイルを読み込みモードで開く (デフォルト)
t	テキストモード (デフォルト)
b	バイナリモード
a	書き込み用で開き、ファイルが存在すれば末尾に追記する

2.2 ファイル操作における例外処理

ファイル操作においてファイル操作中に他の誰かがファイルを操作してしまう、削除してしまうなどという問題が起こってもおかしくありません。そこでファイル操作を確実に終わらせるために例外処理を行いましょう。ここで行う例外処理では先ほど紹介した try と finally を用います。それでは先ほどのプログラムを改良して次のプログラムを実行してみてください。

例外処理

```
a_file=open(”test2.txt”,mode=”w”,encoding=”utf-8”)
try:
    a_file.write(”好きな言葉を入力してください”)
finally:
    a_file.close()
```

このプログラムでは先ほどのプログラムに例外処理を加え確実にファイルを閉じることとしています。仮に try 文が失敗したとしても finally の後のファイルをと知る捜査は確実に行われます。よってファイルを開いたままプログラムが終了すると言ったことは起きなくなります。これで先ほどより安全にファイル操作を行うことが出来ます。

2.3 with 構文

先ほどのように try と finally を書き込むと安全にファイル操作を行うことが出来ますがどうしてもプログラムが冗長になっていしまいます。そこで自動的に処理の最後に close() メソッドを呼んでくれる with 構文を紹介します。

with 構文

with open(ファイル名) as 変数名

このように記述することで with 構文を使用することが出来ます。ここで言う変数名とは先ほどから使っている a_file の代わりになってくれるものです。それでは先ほどのプログラムを with 構文を使って書き換えてみてください。

with

```
with open("test3.txt",mode="w") as f:
   f.write("好きな言葉を入力してください")
```

3 演習

1. 入力された数字が素数か合成数か判断するプログラムを作成せよ。素数の場合は素数、合成数の場合は合成数と表示せよ

解答

```
数字を入力 21
合成数
数字を入力 53
素数
```

2. 入力された数字の数だけアルファベットをランダムで表示するプログラムを作成せよ。乱数発生アルゴリズムは何を使ってもよい

解答

```
数字を入力 8
e
m
d
p
p
h
k
g
```

3. ファイル操作により文字列を入手してその文字列に対して大文字化、小文字化、ハイフンにより文字列の連結をせよ。ファイルは今回の講座で使ったものを使用せよ

解答

```
KEEP ON ASKING, AND IT WILL BE GIVEN YOU:
KEEP ON SEEKING, AND YOU WILL FIND:
KEEP ON KNOCKING, AND IT WILL BE OPENED TO YOU:
Keep on asking and it will be given you
Keep on seeking and you will find
Keep on knocking and it will be opened to you
Keep-on-asking-and-it-will-be-given-you-Keep-on-seeking-and-you-will-find-
Keep-on-knocking-and-it-will-be-opened-to-you
```

4. 文字列を分割し正規表現を再現せよ。今回使う文字列は $adbcacbdexxyyzzabchixyzopqssabexypzxyz$ で正規表現によって次のようにすること, なおこの問題は正規表現がわからない場合は解かなくてもよい

解答

```
(15, 18) abc
(20, 23) xyz
(35, 38) xyz
```