【課題 27：monte_sample.py】

課題 26 と同様に，モンテカルロサンプリングにより，円周率を求めるコードをマルチプロセスにより実装せよ．モンテカルロサンプリングでは，0 以上 1 未満の乱数ペアを多数発生させ，半径 1 の円の内側の点を表す乱数ペア（乱数ペアの 2 乗和が 1 未満のもの）をカウントする．そして，円の内側の点が全乱数ペアのうち何割あったかにより円周率を求める．わからない場合は，`https://manabitimes.jp/math/1182` を参照すること．

今回は，関数の呼び出し時と呼び出し終了時の所要時間を計測する関数を定義し，サンプリングする関数をデコレートすることで各関数の所要時間とプログラム全体の所要時間を出力する．

採点の都合上，以下の要件を満たすこととする：

(1) 関数実行の所要時間を計測する関数デコレータを定義する；
  (a) デコレート対象の関数の属性を引き継ぐ；
  (b) デコレート対象の関数実行直前の時刻を記録する；
  (c) デコレート対象の関数実行直後の時刻を記録し，所要時間を計測する；
  (d) 図 1 のように出力する；
  (e) このデコレータ以外で時間計測および所要時間の出力をしてはいけない；
(2) 所要時間計測の対象は，各 sampling 関数実行時，および，プログラム全体である：
(3) トータルで，100,000,000 個の乱数によるモンテカルロサンプリングを行う；
(4) これを【100 個】のタスクに分割する（前回と個数が異なる点に注意する）；
(5) multiprocessing モジュールの Pool オブジェクトを使用し，同時実行するプロセス数を制御する；
(6) プロセス数は，コマンドライン引数により指定する；
(7) map メソッドにより以下の sampling 関数を実行し，return された結果リストを受け取る；
(8) 結果リストを集計し，円周率を算出する（4 倍しないと円周率にならないことに注意する）；
(9) sampling 関数を定義する；
  (a) 生成する乱数ペアの個数を表すパラメータ num を持つ；
  (b) 横座標 x と縦座標 y のそれぞれを表す乱数を num 回生成する；
  (c) 条件（$x^2 + y^2 < 1$）を満たす乱数ペアをカウントし，集計結果を return する；
(10)「if __name__」以下を修正する必要がある；

```
PS C:\Users\admin\Desktop\ProA2> python .\lec12\kadai27\monte_sample.py 10
所要時間：0.72
所要時間：0.70
所要時間：0.70
所要時間：0.72



所要時間：0.70
所要時間：0.63
所要時間：0.65
314119280.0/100000000 = 3.1411928
所要時間：10.91
```

```
PS C:\Users\admin\Desktop\ProA2> python .\lec12\kadai27\monte_sample.py 1
所要時間：0.32
所要時間：0.35
所要時間：0.33
所要時間：0.34



所要時間：0.32
所要時間：0.33
所要時間：0.35
314182064.0/100000000 = 3.14182064
所要時間：35.48
```

図 1　課題 27 の実行結果

リスト 1　monte_sample.py

```python
import functools
import multiprocessing
import random
import sys
import time

def sampling(num):
    cnt = 0
    for _ in range(num):
        x = random.random()
        y = random.random()
        if x**2+y**2 < 1:
            cnt += 1
    return cnt

if __name__ == "__main__":
    total = 100000000
    num = int(total/100)
    nums = [num for _ in range(100)]
    num_of_processes = int(sys.argv[1])
    pl = multiprocessing.Pool(num_of_processes)
    results = pl.map(sampling, nums)
    print(f"{4.0*sum(results)}/{total} = {4.0*sum(results)/total}")
```

【課題 28：sql_executer.py】

データベースにレコードを登録するためのコンテキストマネージャを実装せよ．コンテキストマネージャでは，SQL 文実行時にエラーが発生した場合でも，データベースとのコネクションを切断するようにする．

採点の都合上，以下の要件を満たすこととする：

(1) データベースへの接続から切断までを管理するコンテキストマネージャクラス SqlExecuter を実装する：
   (a) イニシャライザで，データベースのパス，テーブル名，カラム名のリストを受け取り，インスタンス変数に登録する；
   (b) ラップする処理開始前に，インスタンス変数の値に基づき，データベースへの接続，カーソルオブジェクトの作成，テーブルの作成を実行する．ただし，同名のテーブルが既に存在する場合は，当該テーブルを削除してから作成する；
   (c) ラップする処理終了後に，エラー発生の如何に関わらず，データベースとの接続を切断する；
   (d) 各メソッド，特殊メソッドには，出力が図 2 になるように print 文を追加する；
   (e) insert メソッドで，挿入するレコード群に関するタプルのリストを受け取り，テーブルにレコードを挿入する；
   (f) select メソッドで，選択するカラムリスト，検索条件，表示順を受け取り，該当レコードを検索し，結果タプルのリストを返す；
(2)「if __name__」以下は修正しない；
(3) 入力データのファイルパス，出力データベースのパスはコマンドライン引数により**この順番で**指定する；
(4)「if __name__」以下では，テーブル名，カラム名のリストを定義し，SqlExecuter クラスに指定してコンテキストマネージャオブジェクト「se」を生成している．そして，insert メソッドを用いてポケモンデータをテーブルに挿入し，select メソッドを用いて条件を満たすレコードを検索，結果を出力している；

```
PS C:\Users\admin\Desktop\ProA2> python .\lec12\kadai28\sql_executer.py lec12/data/poke_names.t
xt lec12/data/pokemon.db
lec12/data/pokemon.dbに接続しました
namesを構築しました
レコードを挿入しました
('セレビィ',)
('ホウオウ',)
('ルギア',)
('バンギラス',)
('サナギラス',)
('ヨーギラス',)
正常にSQLが実行されました
切断しました
```

```
PS C:\Users\admin\Desktop\ProA2> python .\lec12\kadai28\sql_executer.py lec12/data/poke_names.t
xt lec12/data/pokemon.db
lec12/data/pokemon.dbに接続しました
namesは既に存在していたので，一度削除してから構築しました
レコードを挿入しました
('セレビィ',)
('ホウオウ',)
('ルギア',)
('バンギラス',)
('サナギラス',)
('ヨーギラス',)
正常にSQLが実行されました
切断しました
```

```
PS C:\Users\admin\Desktop\ProA2> python .\lec12\kadai28\sql_executer.py lec12/data/poke_names.t
xt lec12/data/pokemon.db
lec12/data/pokemon.dbに接続しました
namesを構築しました
レコードを挿入しました
正常にSQLが実行されませんでした：<class 'sqlite3.OperationalError'> no such column: names
切断しました
Traceback (most recent call last):
  File "C:\Users\admin\Desktop\ProA2\lec12\kadai28\sql_executer.py", line 70, in <module>
    rows = se.select(["names"], "id > 245", "id DESC")
  File "C:\Users\admin\Desktop\ProA2\lec12\kadai28\sql_executer.py", line 49, in select
    self.cur.execute(sql)
sqlite3.OperationalError: no such column: names
```

図 2　課題 28 の実行結果：上段は，同名のテーブルが存在しない場合で，エラーなく処理が完了した際の実行結果／中段は，同名のテーブルが存在する場合で，エラーなく処理が完了した際の実行結果／下段は，選択するカラム名がテーブルに存在せず，エラーが発生した際の実行結果

リスト 2 sql_executer.py

```
import sqlite3
import sys

class SqlExecuter:
    def __init__():

    def __enter__():

    def __exit__():

    def insert():

    def select():


def read_names(file_path):
    names = []
    with open(encoding="utf8", file=file_path, mode="r") as rfo:
        for row in rfo:
            id_, tit, typ, *evo_lst = row.rstrip().split("\t")
            evo = " ".join(evo_lst)
            names.append((int(id_), tit, typ, evo))
    return names


if __name__ == "__main__":
    names = read_names(sys.argv[1]) # "lec12/data/poke_names.txt"
    db_path = sys.argv[2] # "lec12/data/pokemon.db"
    tbl_name = "names"
    col_lst = ["id", "name", "types", "evolvs"]
    with SqlExecuter(db_path, tbl_name, col_lst) as se:
        se.insert(names)
        rows = se.select(["name"] condition="id > 245", order="id DESC")
        for row in rows:
            print(row)
```

【課題 29：best_party.py】

課題 08 と同様に，ポケモン 3 体からなるパーティーを構築し，スコアが最も高いパーティーを探すコードをマルチプロセスにより実装せよ．配布した名前ファイル「poke_names.txt」と種族値ファイル「base_stats.txt」を読み込み，全 251 種のポケモンインスタンスを要素とするリストを作成する．そして，251 種から 3 体選ぶ全組合せに対して，スコアを計算する．ポケモンパーティーのスコアは，3 体のポケモンの種族値の合計値とする．

採点の都合上，以下の要件を満たすこととする：

(1) 1 つのポケモンパーティーに対して，スコア（種族値の合計）を計算する party_score 関数を定義する；
(2) パーティーリストからスコア最大のパーティーを求める max_party 関数を定義する；
(3)「if __name__」以下の「ここから」から「ここまで」の間で，以下を実行する；
   (a) itertools モジュールの combinations 関数により，3 ポケモンからなるパーティーリストを生成する；
   (b) これを 251 個のタスクに分割する．すなわち，全 2604125 パーティーを 10375 パーティーからなる 251 個のリストのリストに分割する；
   (c) multiprocessing モジュールの Pool オブジェクトを使用し，同時実行するプロセス数を制御する．プロセス数は，コマンドライン引数により指定する；
   (d) map メソッドにより，サブパーティーごとにスコア最大のパーティーを探索し，結果リストを受け取る；
   (e) 結果リストの中からスコア最大のパーティーを探索し，全パーティーにおける最大パーティーを求める；
   (f) 名前ファイルのパス，種族値ファイルのパス，プロセス数は，コマンドライン引数により，**この順番**で指定する；
   (g) 出力結果が図 3 になるようにする；

```
PS C:\Users\admin\Desktop\ProA2> python .\lec12\kadai29\best_party.py
lec12/data/poke_names.txt lec12/data/base_stats.txt 1
(ミュウツー，ルギア，ホウオウ)
所要時間：3.76
PS C:\Users\admin\Desktop\ProA2> python .\lec12\kadai29\best_party.py
lec12/data/poke_names.txt lec12/data/base_stats.txt 10
(ミュウツー，ルギア，ホウオウ)
所要時間：2.06
```

図 3　課題 29 の実行結果

リスト 3 best_party.py

```
class Monster:
    def __init__(self, title, stats):
        self.title = title
        self.stats = stats

    def __repr__(self):
        return self.title


def read_files(file_path1, file_path2):
    titles = []
    with open(file_path1, "r", encoding="utf8") as rfo:
        for row in rfo:
            _, tit, _, *_ = row.rstrip().split("\t")
            titles.append(tit)

    stats = []
    with open(file_path2, "r") as rfo:
        for row in rfo:
            row = row.rstrip()
            stats.append([int(col) for col in row.split(" ")])

    monsters = [Monster(title, stat) for title, stat in zip(titles, stats)]
    return monsters


if __name__ == "__main__":
    bgn = time.time()
    monsters = read_files(sys.argv[1], sys.argv[2]) # lec11/data/poke_names.txt lec11/data/
        base_stats.txt

    #ここから


    #ここまで

    end = time.time()
    print(f"所要時間：{end-bgn:.2f}")
```