11 Python基礎 補足

練習3

- 以下のモジュールを利用するプログラムを作成したところ、正常に動作しなかった。
    - 原因を突き止めてください。
- 作成したファイル【math.py】

```
from math import sin

x = 45
y = sin(x)
print(y)
```

【実行結果】

```
$ python math.py
Traceback (most recent call last):
  File "/home/yoshimura/src/11/math.py", line 1, in <module>
    from math import sin
  File "/home/yoshimura/src/11/math.py", line 1, in <module>
    from math import sin
ImportError: cannot import name 'sin' from partially initialized module 'math' (most
likely due to a circular import) (/home/yoshimura/src/11/math.py)
```

Linux,MacOSでは仕様通り動作しないが、Windowsでは動作してしまう件

- 確認するためのコード1

```
import math
print(math.__file__)
```

Windows

```
> python check_file1.py
Traceback (most recent call last):
  File "Z:\Dropbox\2025 Trident\zenki\siryo\NT_Python\src\11\py_test\check_file.py",
line 2, in <module>
    print(math.__file__)
AttributeError: module 'math' has no attribute '__file__'
```

Linux

```
$ python check_file1.py
/home/yoshimura/anaconda3/envs/py39/lib/python3.9/lib-dynload/math.cpython-39-x86_64-linux-gnu.so
```

- math の特例挙動（built-in + C extension）
  Python において math は C拡張で実装された built-in module（組み込みモジュール）

- 確認するためのコード2

```
import sys
print('math' in sys.builtin_module_names)
print(sys.builtin_module_names)
```

Windows

```
>python check_file2.py
True
('_abc', '_ast', '_bisect', '_blake2', '_codecs', '_codecs_cn', '_codecs_hk',
'_codecs_iso2022', '_codecs_jp', '_codecs_kr', '_codecs_tw', '_collections',
'_contextvars', '_csv', '_datetime', '_functools', '_heapq', '_imp', '_io', '_json',
'_locale', '_lsprof', '_md5', '_multibytecodec', '_opcode', '_operator',
'_peg_parser', '_pickle', '_random', '_sha1', '_sha256', '_sha3', '_sha512',
'_signal', '_sre', '_stat', '_statistics', '_string', '_struct', '_symtable',
'_thread', '_tracemalloc', '_warnings', '_weakref', '_winapi', '_xxsubinterpreters',
'array', 'atexit', 'audioop', 'binascii', 'builtins', 'cmath', 'errno',
'faulthandler', 'gc', 'itertools', 'marshal', 'math', 'mmap', 'msvcrt', 'nt',
'parser', 'sys', 'time', 'winreg', 'xxsubtype', 'zlib')
```

Linux

```
$ python check_file2.py
False
('_abc', '_ast', '_codecs', '_collections', '_functools', '_imp', '_io', '_locale',
'_operator', '_peg_parser', '_signal', '_sre', '_stat', '_string', '_symtable',
'_thread', '_tracemalloc', '_warnings', '_weakref', 'atexit', 'builtins', 'errno',
'faulthandler', 'gc', 'itertools', 'marshal', 'posix', 'pwd', 'sys', 'time',
'xxsubtype')
```

補足：実装の詳細（CPython）

CPython のモジュールローダーは以下の順序でモジュールを解決する

1. `sys.modules` にキャッシュがあればそれを使用
2. `sys.builtin_module_names` にあれば built-in としてロード
3. `sys.path` に沿って `.py`, `.pyc`, `.pyd`, `.so` を探索

検証

- `sys.builtin_module_names` に含まれないモジュールでテストを行ってみる

```
import datetime

print(datetime.date.today())
```

Windows

```
>python datetime_.py
2025-05-29
```

Linux

```
$ python datetime_.py
2025-05-29
```

- ファイル名を `datetime_.py` → `datetime.py` に変更し実行する
  Windows

```
>python datetime.py
Traceback (most recent call last):
  File "Z:\Dropbox\2025 Trident\zenki\siryo\NT_Python\src\11\py_test\datetime.py",
line 1, in <module>
    import datetime
  File "Z:\Dropbox\2025 Trident\zenki\siryo\NT_Python\src\11\py_test\datetime.py",
line 3, in <module>
    print(datetime.date.today())
AttributeError: partially initialized module 'datetime' has no attribute 'date' (most
likely due to a circular import)
```

Linux

```
$ python datetime.py
Traceback (most recent call last):
  File "/home/yoshimura/Dropbox/2025
Trident/zenki/siryo/NT_Python/src/11/py_test/datetime.py", line 1, in <module>
    import datetime
  File "/home/yoshimura/Dropbox/2025
Trident/zenki/siryo/NT_Python/src/11/py_test/datetime.py", line 3, in <module>
    print(datetime.date.today())
AttributeError: partially initialized module 'datetime' has no attribute 'date' (most
likely due to a circular import)
```

どちらも同じエラーを出力することが解った！

まとめ

- `sys.builtin_module_names` に含まれるモジュールは、Python インタプリタにあらかじめ組み込まれている
- 実行順序（優先度）が高い
- 含まれないモジュールは、標準ライブラリ (pure Python)であり、Python に同梱されるが、 `lib` フォルダ内に `.py` ファイルとして存在する
    - `C:\\Users\\username\\anaconda3\\envs\\py39\\lib` を実際に確認すると、.pyが大量に存在する。（以下一例：datetime.py）

```
"""Concrete date/time and related types.

See http://www.iana.org/time-zones/repository/tz-link.html for
time zone and DST data sources.
"""

__all__ = ("date", "datetime", "time", "timedelta", "timezone", "tzinfo",
           "MINYEAR", "MAXYEAR")


import time as _time
import math as _math
import sys

def _cmp(x, y):
    return 0 if x == y else 1 if x > y else -1

MINYEAR = 1
MAXYEAR = 9999
_MAXORDINAL = 3652059  # date.max.toordinal()

# Utility functions, adapted from Python's Demo/classes/Dates.py, which
# also assumes the current Gregorian calendar indefinitely extended in
# both directions.  Difference:  Dates.py calls January 1 of year 0 day
# number 1.  The code here calls January 1 of year 1 day number 1.  This is
# to match the definition of the "proleptic Gregorian" calendar in Dershowitz
# and Reingold's "Calendrical Calculations", where it's the base calendar
# for all computations.  See the book for algorithms for converting between
# proleptic Gregorian ordinals and many other calendar systems.

# -1 is a placeholder for indexing purposes.
_DAYS_IN_MONTH = [-1, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31]

_DAYS_BEFORE_MONTH = [-1]  # -1 is a placeholder for indexing purposes.
```

```
dbm = 0
for dim in _DAYS_IN_MONTH[1:]:
    _DAYS_BEFORE_MONTH.append(dbm)
    dbm += dim
del dbm, dim

def _is_leap(year):
    "year -> 1 if leap year, else 0."
    return year % 4 == 0 and (year % 100 != 0 or year % 400 == 0)

def _days_before_year(year):
    "year -> number of days before January 1st of year."
    y = year - 1
    return y*365 + y//4 - y//100 + y//400

def _days_in_month(year, month):
    "year, month -> number of days in that month in that year."
    assert 1 <= month <= 12, month
    if month == 2 and _is_leap(year):
        return 29
    return _DAYS_IN_MONTH[month]

def _days_before_month(year, month):
    "year, month -> number of days in year preceding first day of month."
    assert 1 <= month <= 12, 'month must be in 1..12'
    return _DAYS_BEFORE_MONTH[month] + (month > 2 and _is_leap(year))

  ⋮
```