模拟面试参考答案

1. 下面的代码会输出什么。

```
the_list = [1, 2, 3, 4]
print([i for i in the_list if i > 2])
print([i for i in the_list if i % 2])
print({x: f'item{x ** 2}' for x in (2, 4, 6)})
print(len({x for x in 'hello world' if x not in 'abcdefg'}))
```

答：

```
[3, 4]
[1, 3]
{2: 'item4', 4: 'item16', 6: 'item36'}
6
```

2. 有一个通过网络获取数据的函数（可能会因为网络原因出现异常），写一个装饰器让这个函数在出现异常时可以重试指定的次数，并在每次重试之前随机延迟一段时间。

答：

写法一：

```
from functools import wraps
from random import random
from time import sleep


def retry(*, retry_times=3, max_wait_secs=5, errors=(Exception,
)):

    def decorate(func):

        @wraps(func)
        def wrapper(*args, **kwargs):
            for _ in range(retry_times):
                try:
                    return func(*args, **kwargs)
                except errors:
                    sleep(random() * max_wait_secs)
            return None

        return wrapper
```

```
    return decorate
```

写法二：

```
from functools import wraps
from random import random
from time import sleep


class Retry():

    def __init__(self, *, retry_times=3,
                 max_wait_secs=5, errors=(Exception,)):
        self.retry_times = retry_times
        self.max_wait_secs = max_wait_secs
        self.errors = errors

    def __call__(self, func):

        @wraps(func)
        def wrapper(*args, **kwargs):
            for _ in range(self.retry_times):
                try:
                    return func(*args, **kwargs)
                except self.errors:
                    sleep(random() * self.max_wait_secs)
            return None

        return wrapper
```

3. 下面的字典中保存了某些公司今日的股票代码及价格，写出从中找出价格最高和价格最低股票的代码。

```
prices = {
    'AAPL': 191.88,
    'GOOG': 1186.96,
    'IBM': 149.24,
    'ORCL': 48.44,
    'ACN': 166.89,
    'FB': 208.09,
    'SYMC': 21.29
}
```

答：

```
max(zip(prices.values(), prices.keys()))
min(zip(prices.values(), prices.keys()))
```

4. 对上面保存股票代码及价格的字典按价格从低到高进行排序，得到排序后的字典。

   答：

```
{v: k for k, v in sorted(zip(prices.values(), prices.keys()))}
```

5. 有N个人（编号为1~N）围成一圈从编号为1的人开始报数，报数的范围是1~M，报到M的人出列，下一个人重新从1开始报数，直到留下最后一个人。编程求出最后一个人的编号。

   答：

```
N = 10
M = 5

persons = [True] * N
total, idx, num = N, 0, 0
while total > 1:
    if persons[idx]:
        num += 1
        if num == M:
            persons[idx] = False
            total, num = total - 1, 0
    idx += 1
    idx %= N
for idx, person in enumerate(persons):
    if person:
        print(idx + 1)
        break
```

6. 编写一个函数，从传入的列表中找出第二大的元素，不允许使用排序和嵌套的循环。

   答：

```
def second_max(items: list):
    assert len(items) >= 2
    first, second = (items[0], items[1]) if items[0] > items[1]
\
        else (items[1], items[0])
    for idx in range(2, len(items)):
        if items[idx] > first:
            first, second = items[idx], first
        elif second < items[idx] < first:
            second = items[idx]
    return second
```

7. 写一个函数，传入的参数是一个列表（列表中的元素可能也是一个列表），返回传入的列表有多少层嵌套。

   答：

```
def list_depth(one_list: list) -> int:
    if isinstance(one_list, list):
        depth = 1
        for elem in one_list:
            sub_depth = list_depth(elem) if isinstance(elem,
list) else 0
            depth = max(depth, sub_depth + 1)
        return depth
    return 0
```

8. 由1、10、100、1000、……组成的序列1101001000……，求这个序列的第N位是0还是1。

   > *输入说明：输入第一个数字（1~10000）表示后面用作测试的数据的数量；从第2个数字开始输入的数字是代表序列第几位的测试数据（1~100000000）。*
   >
   > *输出说明：如果序列中对应的位置是1就输出1，对应的位置是0就输出0.*
   >
   > *例如：*
   >
   > *输入：3 1 2 3*
   >
   > *输出：1 1 0*

   答：这里使用的方法类似于“桶排序”算法，用空间换取时间。

```
def main():
    one_zero_list = [0] * 100000000
    one_zero_list[0] = 1
    index, step = 0, 1
    while True:
        index += step
        if index >= len(one_zero_list):
            break
        one_zero_list[index] = 1
        step += 1
    num, *poses = input().split()
    for index in range(int(num)):
        print(one_zero_list[int(poses[index]) - 1], end=' ')
```

9. 编写一个能够生成斐波拉切数列的迭代器，请说明迭代器的使用方式。

   答：

```python
class Fib:

    def __init__(self, num):
        self.num = num
        self.a, self.b = 0, 1
        self.idx = 0

    def __iter__(self):
        return self

    def __next__(self):
        if self.idx < self.num:
            self.a, self.b = self.b, self.a + self.b
            self.idx += 1
            return self.a
        raise StopIteration()


if __name__ == '__main__':
    fib = Fib(20)
    print(next(fib))
    print(next(fib))
    for val in fib:
        print(val)
```

10. 请阐述什么是跨站脚本攻击（XSS）、跨站身份伪造（CSRF）、点击劫持攻击和SQL注射攻击以及如何防范。

    答：略。

11. 举例说明你在项目中使用过Redis的哪些数据类型?

    答：

    - 字符串类型：页面缓存、查询结果缓存、计数、共享Session、限速。
    - 哈希类型：查询结果缓存、将关系数据库的表映射到内存中。
    - 列表类型：消息队列。
    - 集合类型：维护标签（例如：给用户贴标签、根据标签找用户）。
    - 有序集合类型：排行榜。

12. 请说明你在没有IDE的情况下如何调试Python代码?

    答：Python内置了交互式调试器，可以通过引入pdb模块，并调用`set_trace()`函数来触发调试器，让代码停在指定的位置，然后通过输入局部变量的名称来打印它们的值，或通过`locals()`函数来列出所有的局部变量，当然还可以引入模块、创建新对象、调用`help()`函数获取帮助等。当然我们也可以使用下面的命令让代码继续向下执行：

    - step：执行当前代码并将程序运行到下一条可执行语句开头处，遇到函数会进入函数并停留在函数开头的地方。
    - next：执行当前代码并将程序运行到下一条可执行语句开头处，遇到函数会调用函数并得到结果不会进入函数。

- return：继续运行程序直到函数的return语句处。
- continue：继续运行程序直到下一个断点或`set_trace()`调用点。

13. 请说明在一个新项目中如何重建已有项目中使用的依赖关系。

    答：

    ```
    pip3 freeze > requirements.txt
    pip3 install -r requirements.txt
    ```

14. 在Django项目如何读取配置文件中的信息？如果配置了Redis作为缓存服务，如何获得StrictRedis对象直接操作Redis？如何执行原生SQL操作？

    答：

    ```
    from django.conf import settings
    ```

    ```
    from django_redis import get_redis_connection
    ```

    ```
    from django.db import connection
    ```

15. 请阐述自己参与的项目中有没有使用过多进程或多线程？如何处理线程之间的数据竞争？如何协调多个进程或多个线程之间的工作？

    答：耗时间的任务基本上都用异步消息队列或者多线程的方式进行处理，避免请求被阻塞，在爬虫开发的时候没有相关性的任务可以使用多进程，来发挥CPU的多核特性。线程间的数据竞争可以通过“锁机制”来解决，Python的threading模块内置了Lock类。协调多个线程和多个进程都可以使用队列，一个是`queue`模块的`Queue`，一个是`multiprocessing`模块的`Queue`，前者用于多线程，它是一个可以指定缓冲区大小的阻塞队列；后者用于多进程，它通过管道以及锁和信号量机制来协调多个进程。