<https://github.com/aosabook/500lines> 是一个26k star的开源项目，假设我们现在要fork其中的爬虫项目（https://github.com/aosabook/500lines/tree/master/crawler）进行部署和修改。

Q1:

请问下面哪些行为造成了软件的变化，从而使得爬虫c需要进行软件维护？

a 原有的爬虫不存在的功能

b 需要在原爬虫不支持的环境上部署

c 原有爬虫的错误

d 需要支持新的硬件设备

（正确答案：a, b, c, d )

Q2:

当我修改这个爬虫软件一些抓取的性能障碍时候获知让爬虫在原来不支持的环境也可以运行，我的这种行为算什么？

a 软件集成

b 软件维护

c 软件实现

d 软件测试

（正确答案： b)

Q3：(简单题）

当我发现爬虫里面的代码缺陷和错误，并且修复错误，这种行为属于：

a 预防性维护

b 正确性维护

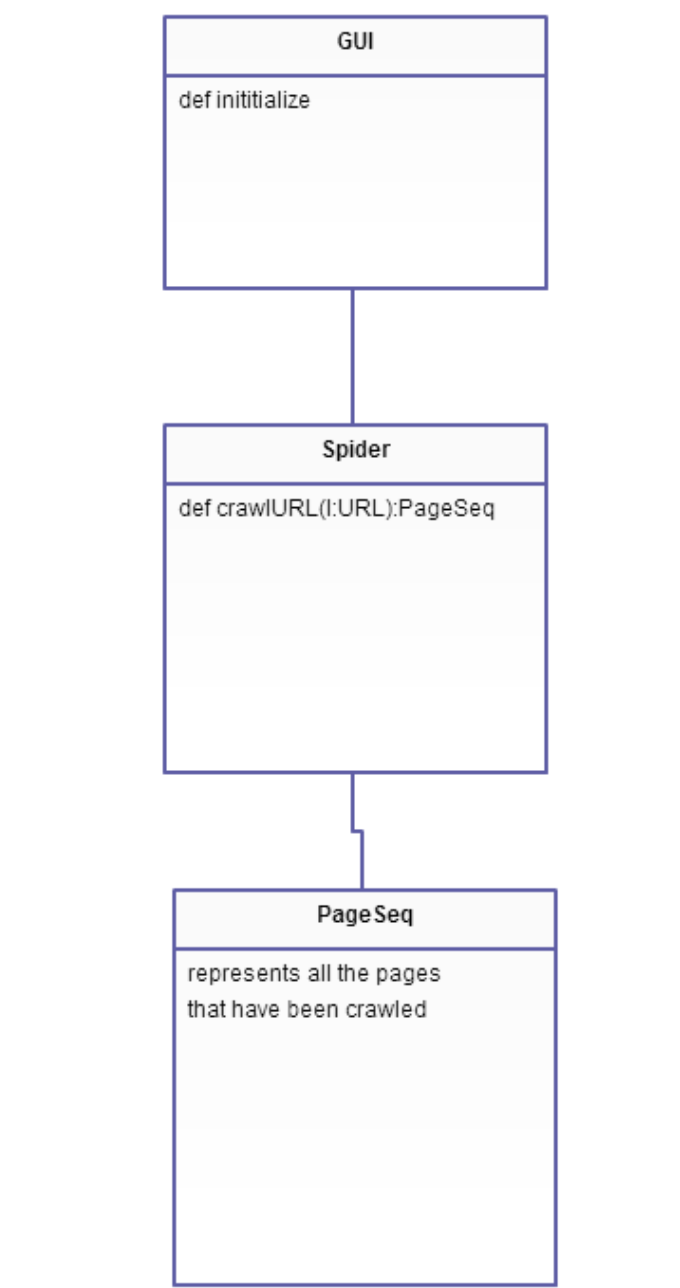
c 适应性维护

d 完成性维护

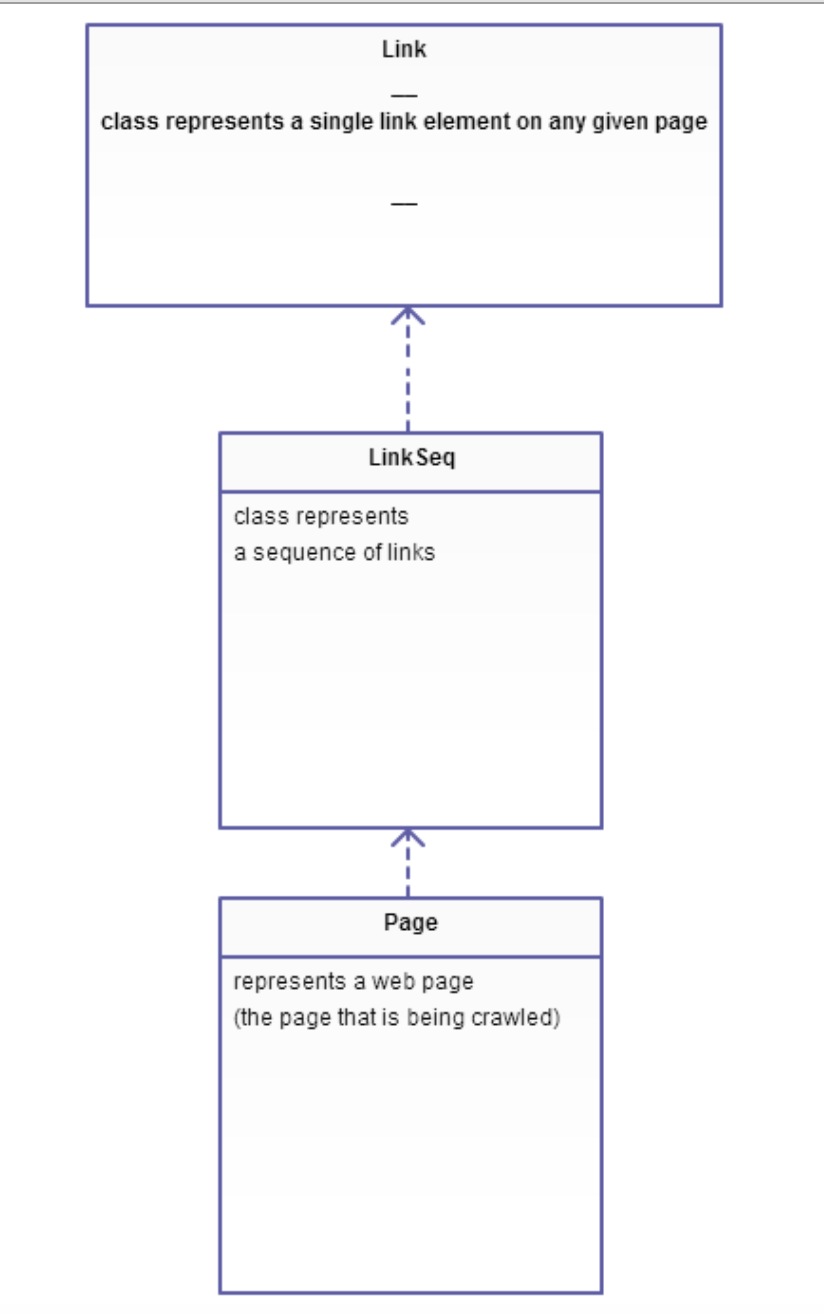
(正确 b）

Q4:

请你画出从GUI界面inital初始化， 到爬虫（Spider)开始crawlURL，最后不同页面按规定顺序PageSeq抓取的爬虫类图

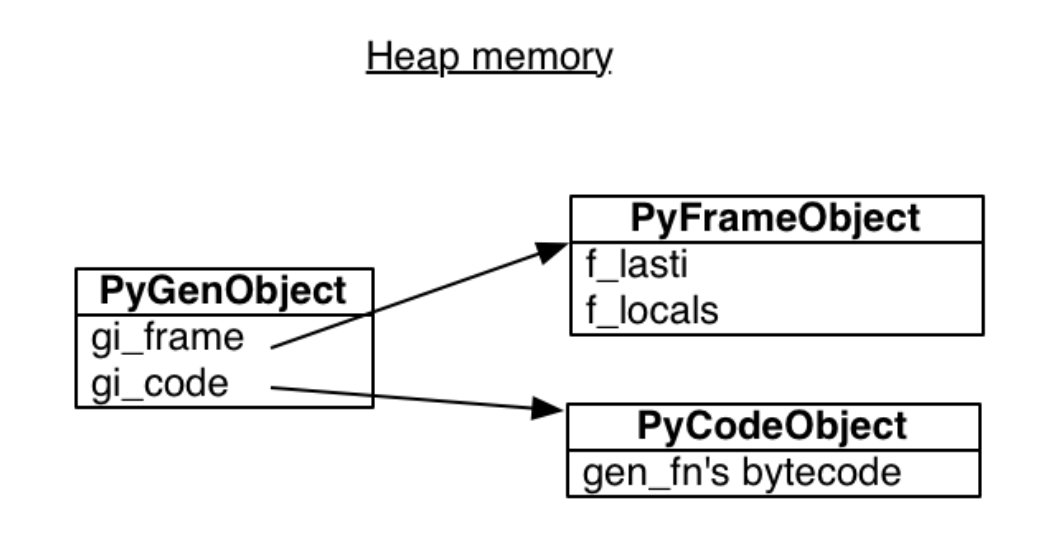


Q5: 如果我们使用Link表示所有的给定网页上的链接元素、LinkSeq表示link的序列，Page表示网页的实体，请你画出相关类图



Q6：(简单题）

查看下面的UML图，了解Python堆栈帧是在堆内存中分配的，这一点至关重要！ Python解释器是普通的C程序，因此其堆栈框架是普通的堆栈框架。 但是它操纵的Python堆栈框架在堆上。 除其他惊喜外，这意味着Python堆栈框架可以超过其函数调用的寿命。 要以交互方式查看此图像，请从栏中保存当前帧。



那么执行下面代码：

>>> import inspect

>>> frame = None

>>> def foo():

... bar()

...

>>> def bar():

... global frame

... frame = inspect.currentframe()

...

>>> foo()

>>> # The frame was executing the code for 'bar'.

>>> print(frame.f\_code.co\_name)

(a)

>>> # Its back pointer refers to the frame for 'foo'.

>>> caller\_frame = frame.f\_back

>>> print(caller\_frame.f\_code.co\_name)

(b)

(a), (b)处应该输出什么？

(答案：a. ’bar’ b.’foo’）