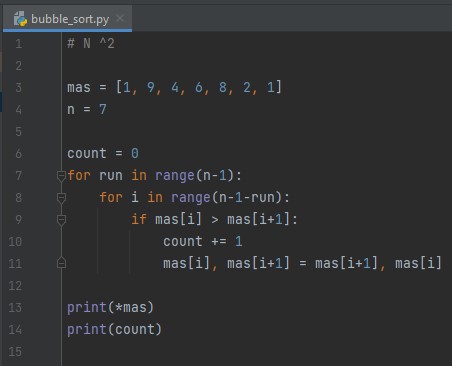
1. какой текст должен присутствовать в названии тестовых файлов и функций. Требование библиотеки Pytest.
   1. “main” **b.** “test\_” **c.** “\_” **d.** “.py”
2. Название файла, который требуется для корректной работы библиотеки **pytest**
   1. chromedriver.exe **b.** test\_visible.py **c.** conftest.py **d.** readme.md
3. Какой будет результат теста?



* 1. Тест пройден **b.** Тест провален **c.** Тест пропущен

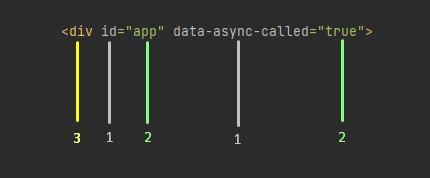
1. что сделает команда ***git commit -m "first commit"***
   * 1. отправляет изменения в репозиторий.
     2. создает коммит с указанным комментарием.
     3. инициализирует локальный **git** репозиторий.
2. В приведенном примере представлена реализация популярного алгоритма “сортировка пузырьком”.

Какая строчка кода содержит оператор условия?



* 1. № 7 **b.** № 13 **c.** № 8 **d.** № 11 **e.** № 9

1. что такое локатор?
   * 1. метод инициализирующий класс
     2. доп строка с типом, ставится после атрибута, указывая на тип которым должен быть этот атрибут.
     3. строка, идентифицирующая элемент **DOM**
     4. свойство элемента **DOM**
2. выберите локатор типа **xpath**
   1. #section **b.** //div/div/h1 **c.** div > div.home-body **d.** div:nth-child(1)
3. Расставьте соответствие в строении элемента **DOM**



|  |  |
| --- | --- |
| 1. | a |
| 2. | d |
| 3. | c |

* + - 1. Имя элемента
      2. Свойство элемента
      3. Значение атрибута
      4. Имя атрибута

1. Что делает Комбинатор **'>'**, в локаторах типа **CSS\_SELECTOR**?
   * 1. выбирает только те элементы, которые являются дочерними непосредственно по отношению к указанному элементу.
     2. выбирает элементы, которые находятся внутри указанного элемента (вне зависимости от уровня вложенности).
     3. выбирает элементы, которые находятся на этом же уровне вложенности, после указанного элемента, с тем же родителем.
     4. выбирает элемент, который находится непосредственно после указанного элемента, если у них общий родитель.

**10.**какие локаторы, из указанных, являются эквивалентными между собой?

* + 1. button#size = button[id=”size”]
    2. button.login = button[text=”login”]
    3. #app = [data-async-called="app"]

**11.** Чем статический метод, в классе, отличается от обычного?

1. К статическому методу можно обратиться не создавая экземпляр класса.
2. В статический метод, в отличии от простого, первым аргументом надо посылать self
3. Статические методы выделяются декоратором @classmethod

**12.**У Вас есть три класса. В какой Вы разместите метод который выполняет следующую функцию:

* + *Выбирает переданное значение в указанном селекте.* Функция универсальная, срабатывает для любого селекта.

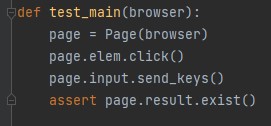
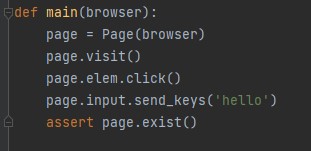
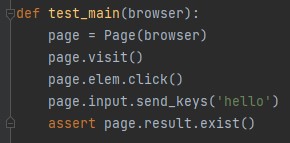
1. **WebElement** - в классе собраны методы и атрибуты элементов DOM
2. **BasePage** - в классе собраны методы и атрибуты любой страницы
3. **Page(BasePage)** - в классе собраны методы и атрибуты страницы Page

**13.**Имеется тест кейс:

* + зайти на страницу **Page**
  + нажать на элемент **elem**
  + ввести текст **‘hello’** в элемент **input**
  + проверить наличие элемента **result**

Выберите блок кода, который реализует этот кейс. Библиотека запуска pytest.

**a. b. c.**



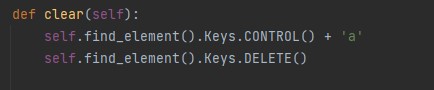
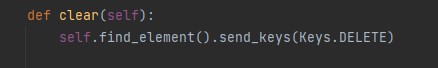
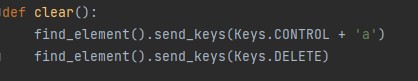
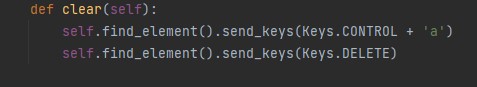
**14.**Как называется метод позволяющий выполнять синхронный **JavaScript** на странице?

**a.** .elem\_javascript() **b.** .get\_dom\_attribute() **c.** .execute\_script() **d.** .set\_window\_size()

**15.**Вам необходимо реализовать метод, который очищает весь текст в заданном элементе.

Выберите подходящую реализацию.

**a.**



**b.**

**c.**

**d.**

**16.**Выберите метод который устанавливает размеры тестируемого окна.

* 1. .window\_size(width=W, height=H)
  2. webdriver.Chrome(width=W, height=H)
  3. .set\_window\_size(width=W, height=H)
  4. driver.window(width=W, height=H)

**17.**что делает атрибут ***driver.current\_url*** ?

* 1. Возвращает результат сравнения текущего урла с переданным
  2. Возвращает текущий URL
  3. открывает переданный URL в новой вкладке

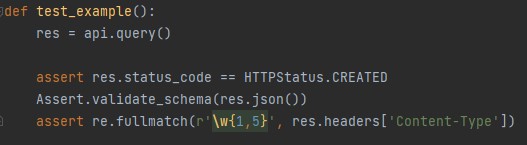
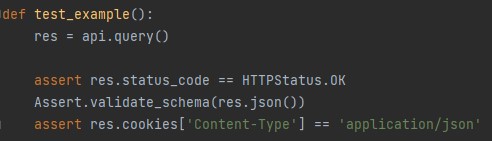
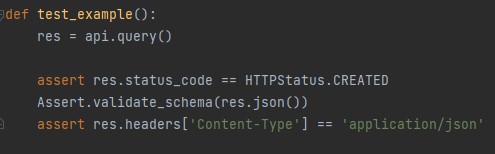
**18.**Выберите декоратор библиотеки pytest, с помощью которого можно классифицировать тесты.

**a.** @pytest.mark.skip **b.** @classmethod **c.** @pytest.mark

**19.**Имеется тест кейс:

* послать запрос - метод query
* Проверить статус ответа - 201
* Проверить валидность ответа
* Проверить заголовок ответа - 'Content-Type': 'application/json'

Выберите блок кода, который реализует этот кейс. Библиотека запуска pytest. **a.**



**b.**

**c.**

**20.**Напишите регулярное выражение под любую дату формата ’12 Августа 2023’

(0[1-9]| [12][0-9]|3[01])\s(Январь|Февраль|Март|Апрель|Май|Июнь|Июль|Август|Сентябрь|Октябрь|Ноябрь|Декабрь)\s(19|20)[0-9][0-9]

Простенько конечно… много пропусков для нереальных дат, но не думаю что в ответе ожидается что-то более детальное)