**Переустановка/обновление приложений (интерфейс windows10):**

1)Заходим Параметры->Приложения и возможности, удаляем старые версии обновляемых программ:

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |

2)Загружаем и устанавливаем с сайта python.org новую версию, выбрав режим с настройкой

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |

3)Загружаем PyCharm с адреса <https://www.jetbrains.com/pycharm/download/?section=windows> и устанавливаем:

|  |  |  |
| --- | --- | --- |
|  |  |  |

4)Создадим проект project\_django в PyCharm, правильно выбрав Virtual environment и Base interpreter (какое-то время будут устанавливаться связи интерпретатора и PyCharm), далее перейдем в окно терминала:

|  |  |  |
| --- | --- | --- |
|  |  |  |

В окне терминала увидим путь к текущему каталогу: C:\Users\leomt\PycharmProjects\project\_django>

5)Активируем виртуальное окружение. Для этого через окно терминала добавим к переменной окружения Path путь к каталогу C:\Users\leomt\PycharmProjects\project\_django\venv\Scripts\ нашего проекта:

a) перейдем в режим командной строки: **cmd.exe**

b)введем: ***set*** *path=%path%; C:\Users\leomt\PycharmProjects\project\_django\venv\Scripts\*

Теперь введем в консоли: **activate.bat**, а затем установим django: ***pip*** *install Django*

Сохраним связи в файл requirements.txt: *pip freeze > requirements.txt*

|  |  |  |
| --- | --- | --- |
|  |  |  |

6)Создадим проект my\_project :

* находясь в папке project\_django введем команду: ***django-admin*** *startproject my\_project;*
* зайдем в созданную на предыдущем шаге папку проекта *my\_project:* ***cd*** *my\_project,* получим:

C:\Users\leomt\PycharmProjects\project\_django\my\_project>

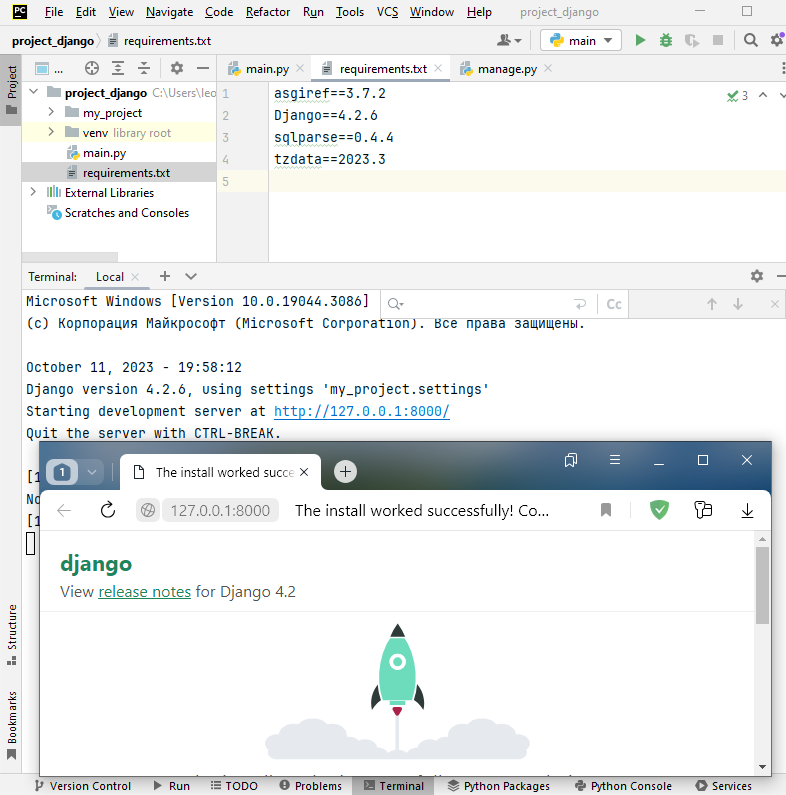
В этой папке есть файл manage.py, с помощью которого запустим сервер: ***python*** *manage.py runserver*

После этого загрузится сервер с адресом <http://127.0.0.1:8000>, запустим и его:

|  |  |  |
| --- | --- | --- |
|  | **Пункты 4-6 представляют собой практическую работу №4.2.1:** |  |

**Перечень действий по пунктам 4-6 представлят собой практикум №1 модуля 4.2 (практикум 4.2.1**

Отчетом по работе 4.2.1 может быть скриншот, подобный предствленному ниже:



**Пример отчета по заданию 1 блока 4.2**

Из рисунка видно по requirements.txt, что django установлен, по окну терминала, что сервер запущен, а из фрагмента браузера - что django работает. **Скриншот (индивидуальный у каждого) - отчет по заданию 1.**

**Задание 2 удобно выполнять сразу после первого.**

1)Перейдем в папку project\_django (на один уровень вверх): **cd ..**

Системное приглашение будет иметь вид: C:\Users\leomt\PycharmProjects\project\_django>

2)Для создания нового проекта из шаблонов нам необходимо запустить ***python*** *manage.py startapp hello,* но это вызовет ошибку, поскольку в текущем каталоге нет файла *manage.py.* Изменим командную строку с учетом расположения данного файла: ***python*** *my\_project\manage.py startapp hello*

Теперь всё успешно – проект hello создан, но нужно будет в папке my\_project найти файл settings.py, где в список *INSTALLED\_APPS* следует добавить *'hello',* после чего проект будет знать о существовании *hello.*

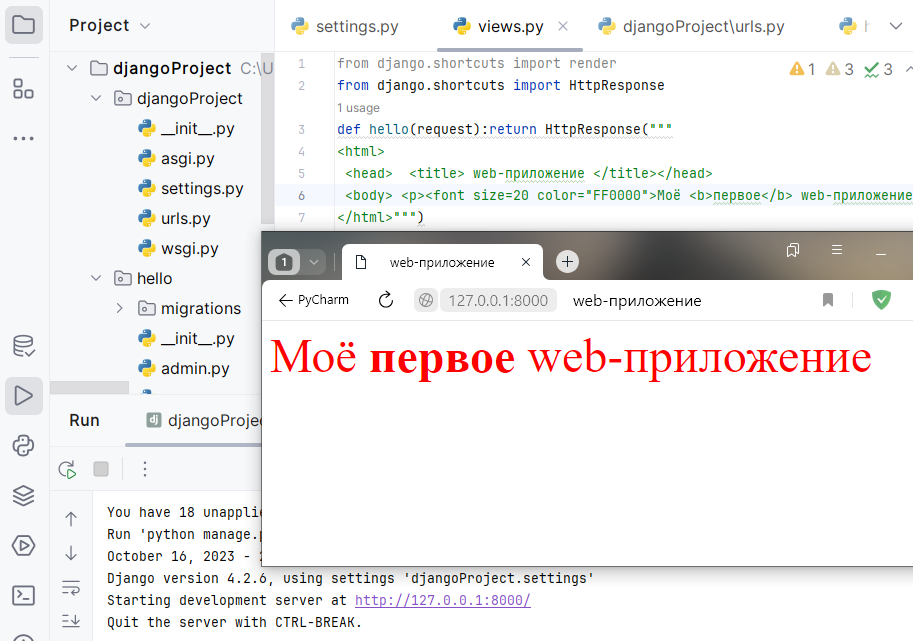
3)Для отображения страницы пропишем ее HTML-код в файл *views.py* папки *hello,* а маршруты допишем в файл *urls.py* папки *my\_project*:

|  |  |
| --- | --- |
| *from django.shortcuts import HttpResponse*  *def hello(request):return HttpResponse("""*  *<html><head> <title> web-приложение </title></head>*  *<body> <font size=20 color=”FF0000”>Моё <b>первое </b>*  *web-приложение </body></html>""")* | *from django.contrib import admin*  *from django.urls import path, include*  *urlpatterns = [*  *path('admin/', admin.site.urls),*  *path('', include('hello.urls'))]* |

Функция *hello* принимает аргумент request. Это создаст HttpRequestObject при загрузке страницы, содержащий информацию о запросе - метод, который может принимать несколько значений, включая GET и POST.

|  |  |
| --- | --- |
| При переходе к приложению, вызовется *hello/urls.py* и регистрирует URL-адреса, определенные там.  Теперь создаем *hello/*urls.py*,* со следующим кодом: | *from django.urls import path*  *from hello import views*  *urlpatterns = [path('', views.hello, name='hello')]* |

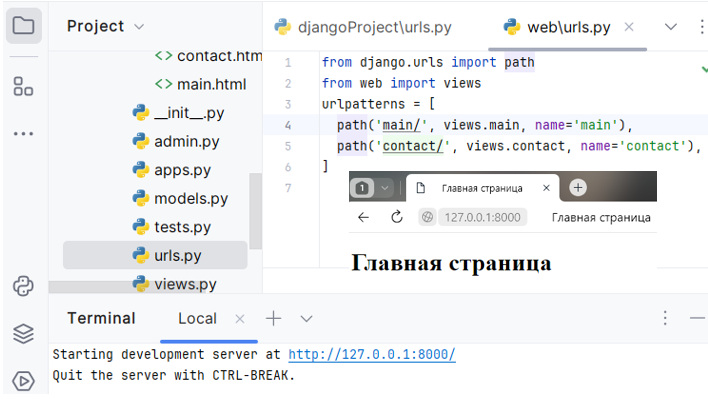
В терминале или с консоли запустим локальный сервер: ***python*** *manage.py runserver,* после чего запустим страницу по ее IP-адресу. Этот алгоритм действий хорошо работает, если запускать всё откуда угодно, но не из Windows в среде PyCharm Community, где на последнем этапе идет рассогласование адресов при запуске сервера, которое не совсем ясно, как устранить. Проще установить 30-дневный PyCharm Professional, где всё работает:



**Скиншот, подобный приведенному выше, принимается как отчет по практической работе 4.2.2.**

Теория и порядок выполнения практической работы **4.2.3** дан на страницах 9-12 файла 4\_2.doc, ссылка на который предшествует ссылке на данный файл (на сайте - ниже).

**Если всё делать в PyCharm Professional, то проблем нет и отчет по практике 4.2.3 может выглядеть так:**



Теория и порядок выполнения практической работы **4.2.4** дан на страницах 13-15 файла 4\_2.doc, ссылка на который предшествует ссылке на данный файл (на сайте - ниже).

**Если всё делать в PyCharm Professional, то проблем нет и отчет по практике 4.2.4 может выглядеть так:**

|  |  |  |  |
| --- | --- | --- | --- |
| 127.0.0.1:8000/web/main Здесь картинка – любой скриншот с именем wallpaper.jpeg |  | 127.0.0.1:8000/web/contact | Скриншот проекта (фрагмент) |

Теперь подробно остановимся на блоке 4.3. Здесь в предоставленных в лекциях фрагментах кода содержится большое количество ошибок и пояснений, допускающих многозначность ответов. Еще следует обратить внимание на неверные отступы в текстах Python-модулей, что приводит к потере их работоспособности.

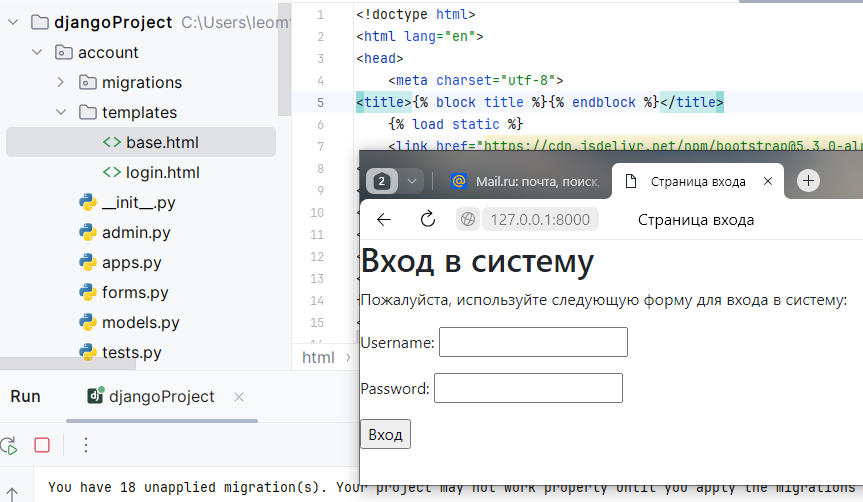
В блоке лекций 4.3, размещенных на smtl.ru/leo, все ошибки уже исправлены, а здесь добавлены некоторые пояснения по этому поводу и примеры форм отчетности.

1.В блоке заданий 4.3.1 в пункте 4) добавлено пояснение, что файл base.html нужно создать в папе account. Кроме того, на рис.1 показано, что в папке account уже есть папка templates и файл forms.py, хотя они будут созданы позже – в п.5) и 6).

2.В п.7) важно, чтобы строка 6 кода совпадала по отступам со строкой 16, строки 7 и 8 были одинаково смещены относительно 6-й, а строки 9-11 и 15 одинаково смещены относительно 8-й, строки 12 и 14 – относительно11-й, а строка 13 – относительно 12-й. В исходном тексте лекций это не соблюдается и ветвление не работает.

3.В account/urls.py неправильно был обозначен путь к функции - исправлено.

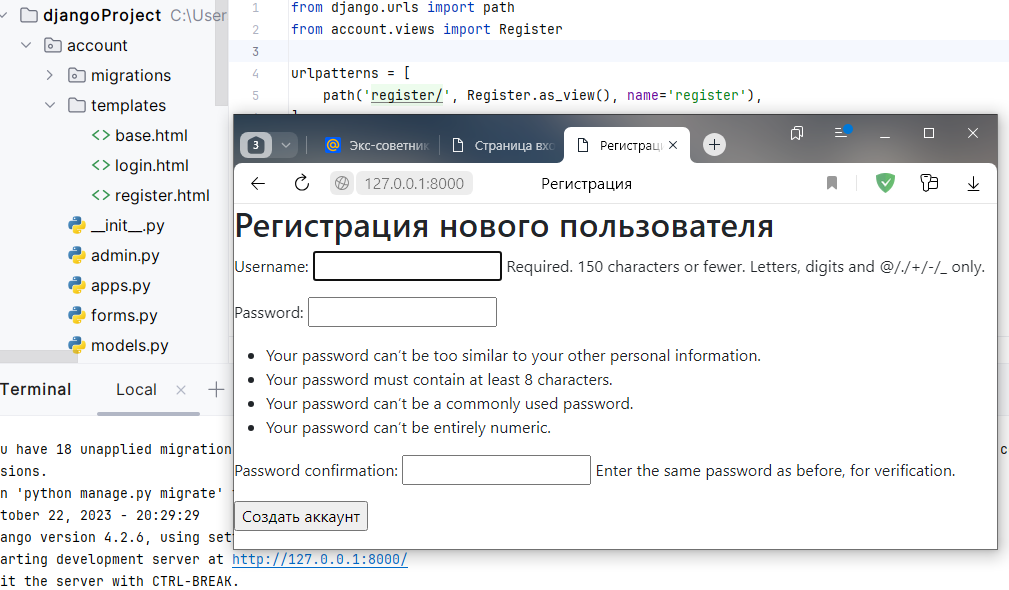
Ниже представлен скриншот – пример отчета по работе.



Переходим к практике 4.3.2. Здесь есть сразу в нескольких местах, но одинаковый тип ошибки – указано, что нужно создать файл с неким содержимым, но *на самом деле, если такой файл уже есть, то предлагаемое содержимое к нему нужно добавить!* В лекциях скорректированных всё это уже исправлено, просто нужно быть внимательным.

Особо хочу отметит ошибки в п.6), где нужно не забыть правильно импортировать класс Register: *from account.views import Register* и убрать лишние строки из *urlpatterns.*

Отчет по практике 4.3.2. примерно соответствует представленному ниже скриншоту:



Наиболее трудоемкой является практика 4.3.3, содержащая 13 пунктов и много ошибок, но после внесенных исправлений проблем нет, особенно если в п.5) рис.8 назначить новому пользователю Staff Status, что настоятельно рекомендуется сделать. Только тогда обычный пользователь получит доступ к редактированию своего профиля (рис.14).

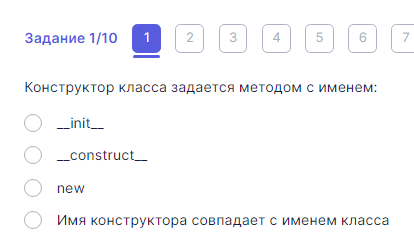
Отчет по 4.3.3 следует представить в нескольких скриншотах, заархивировав их в 1 файл, как это показано в файле 4\_3.docx в лекциях на моем сайте.

Для заданий п. 4.4 необходимо пользоваться лекциями 4\_4.docx с сайта.

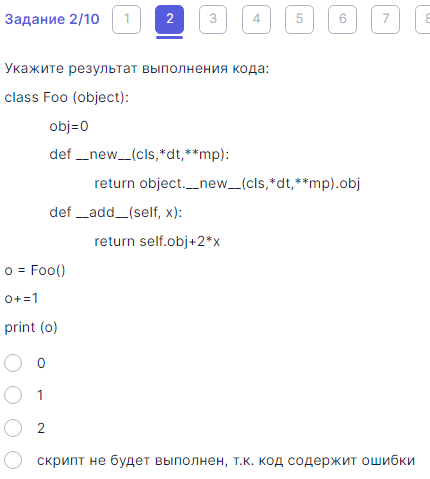
**Справка при ответе на вопросы тестирования по 4 модулю+итоговый тест**

**Внимание!!!** При ответе на вопросы задания, если использовать копирование, то изменяются отступы и программа становится неработоспособной!

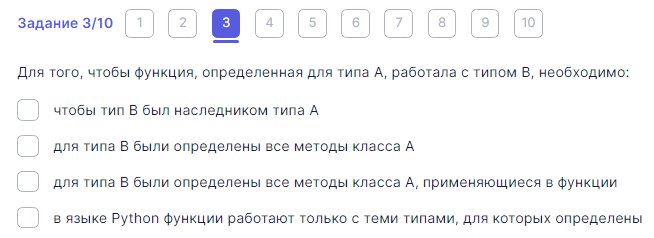
Комментарии к тестам (тесты за 4 модуль различаются)



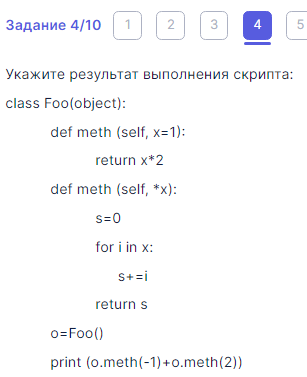
1 (\_\_init\_\_) - правильно



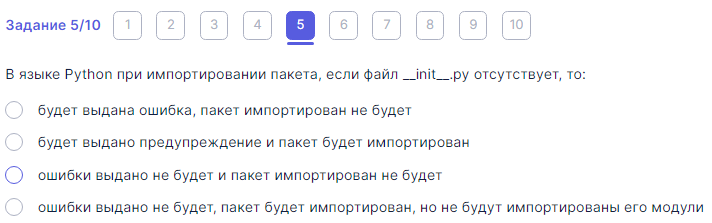
2 – й пункт (ответ «1»)



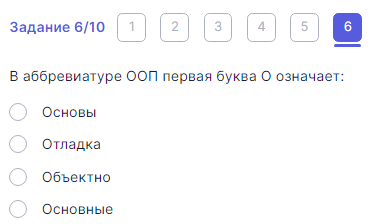
1 (ошибка). **Скорее всего, правильные ответы 1, 2**



Правильный ответ: 1 (здесь случайно забыли сместить две последние строки влево)



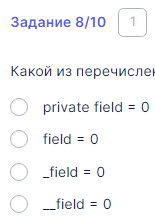
4 (ошибка) – не знаю, какой ответ верный (реально – 4-правильно). **Может быть 1-й считается правильным – до версии Питона 3.2 было так, как в п.1**



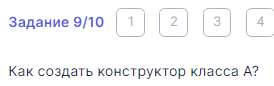
3



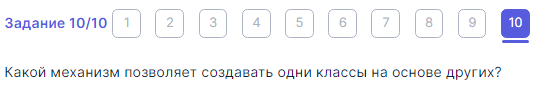
1



4



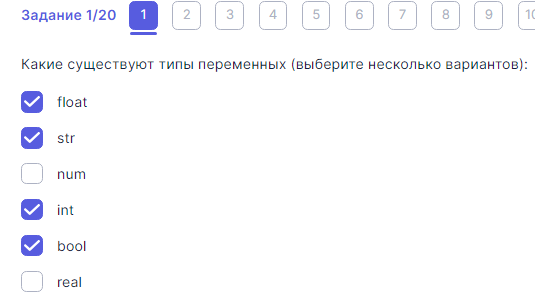
def \_\_init\_\_(параметры конструктора) (ошибка, не знаю, что имеется ввиду)

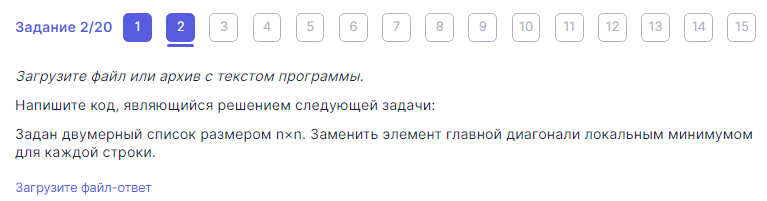


Наследование

Дальше – итоговый тест…

**Переходим к итоговому тестированию – одинаковому у всех**





from random import randint as rnd

n=4

x=[]

for i in range(n):

t=[]

for j in range(n):

t+=[rnd(5,25)]

x+=[t]

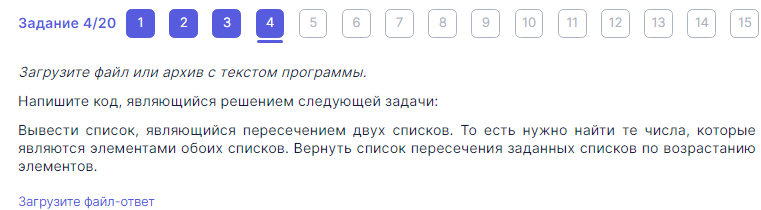
print(x)

for i in range(n):

t=min(x[i])

x[i][i]=t

print(x)



from random import randint as rnd

n=13;m=12;x=[];y=[];z=[]

for i in range(n): x+=[rnd(5,25)]

for i in range(m): y+=[rnd(5,25)]

print(x);print(y)

if n<m:

for i in range(n):

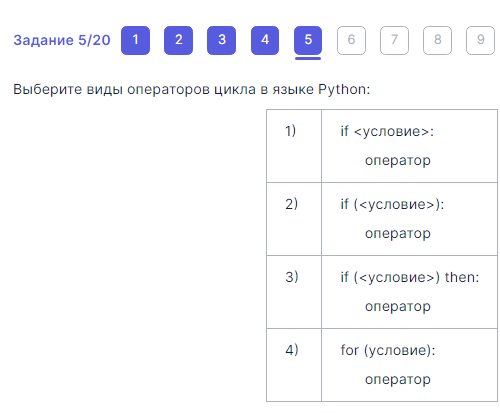
if x[i] in y: z+=[x[i]]

else:

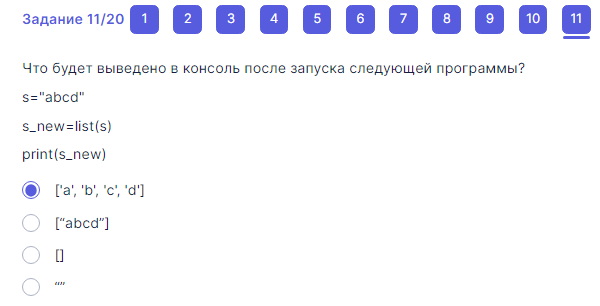
for i in range(m):

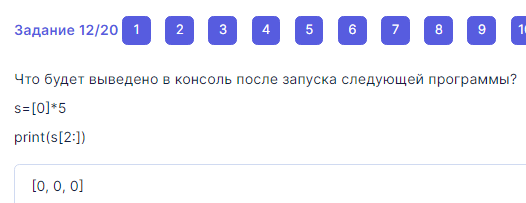
if y[i] in x: z+=[y[i]]

print(sorted(z))

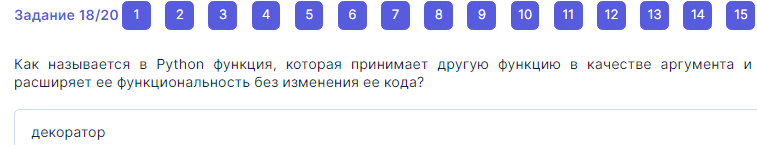


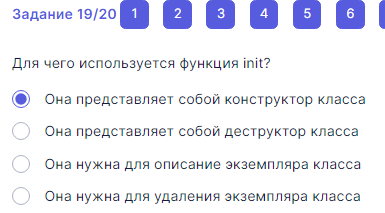
**1,2 -правильно** (бред – верных ответов нет, цикл – только for, но там не условие)





**Без пробелов? С пробелами считают ответ ошибочным, хотя так выдает компьютер**





Вопрос 20 итогового теста

Загрузите на платформу единый архив с материалами, полученными по итогам выполнения проекта «Администрирование веб-сайта»:

1. Учетный лист проекта.
2. Листинг программы, реализующий веб-приложение по выбранной теме.
3. Презентацию проекта.
4. Скриншот разработанного приложения.

Если ничего этого нет, то нужно загрузить doc-файл с текстом, типа: «не хватило времени для реализации этого задания», и завершить тестирование. Дальше я проверяю практическую часть и п.20, оцениваю, а после только можно будет проверить результат.