

python hard oop

Всего 7/7

Множественное наследование, дескрипторы,
метаклассы, слоты





Как динамически создать
такой же класс? *

1 из

1



```
class A(list):  
    pass
```



A = type('A', (list,), {})



A = type('A', (object,), {'type': 'list'})



A = type(list, (object,), {'type': 'A'})



A = list('A', (object,), {})



✓ По какому алгоритму 1
определяется порядок поиска из
методов при множественном 1
наследовании? *

☐ Не по какому, т.к. множественное наследование запрещено

☐ Формула Неймана

☐ Метод Байеса

☐ Выбирается всегда родительский класс указанный первым



☒ С3 - линеаризация



✓ Что будет выведено в терминал? *

1 из

1

```
 class A:  
     pass
```

```
|
```

```
a = A()  
a.b = 10  
print(a.b)
```

☒ 10



☐ None

☐ Будет ошибка

☐ b

☐ a b



✓ Класс является дескриптором 1 из
если * 1

☐ В нем определен метод `__describe__`

☒ В нем определен хотя бы один из методов `__get__`, `__set__`, `__delete__` ✓

☐ В нем определен метод `__get__`

☐ Он является метаклассом



✓ Что будет выведено в терминал? *

1 из

1

```
class A:  
    __slots__ = ('a', 'b')
```

```
a = A()
```

```
print(a.__dict__)
```

☒ Будет ошибка



☐ ('a','b')

☐ {'a': None, 'b': None}

☐ __dict__

☐ None





Что будет выведено в терминал? *

1 из

1

```
class A:  
    pass  
  
a = A()  
print(a.__class__, A.__class__, a.__class__.__class__, a.__class__.__class__.__class__)
```



<class '__main__.A'> <class 'type'>
<class 'type'> <class 'type'>



<class '__main__.A'> <class '__main__.A'>
<class '__main__.A'> <class 'type'>



<class '__main__.A'> <class 'type'> <class
'type'> <class 'object'>



<class '__main__.A'> <class 'object'>
<class 'type'> None



✓ Чтобы узнать порядок разрешения методов, который в данном случае принял Python (при множественном наследовании) можно * 1 из 1

☒ посмотреть значение атрибута `__mro__` ✓

☐ посмотреть значение атрибута `__slots__`

☐ посмотреть значение атрибута `__dict__`

☐ использовать функцию `dir`

☐ подбросить монету



Google Формы

