

Объектно-ориентированное программирование

Семинар 1





План курса

Принципы ООП: Инкапсуляция, наследование, полиморфизм
Принципы ООП Абстракция и интерфейсы. Пример проектирования
Некоторые стандартные интерфейсы Java и примеры их использования
ООП: Обобщения
От простого к практике
ООП Дизайн и Solid
Есть ли жизнь без Java?



Цели на семинар:

- повторить теорию базовых определений ООП – абстракция, инкапсуляция, наследование, полиморфизм
- получить практические навыки в написании сущностей и логики согласно принципам ООП
- научить составлять идеологически верный ООП программы



Инкапсуляция





Инкапсуляция - Задание

```
public class Product{  
    String name;  
    double cost;  
}
```

```
public class VendingMachine{  
    ArrayList<Product>;  
    Product getProductByName();  
    Product getProductByCost();  
}
```





Наследование





Наследование - Задание

```
public class Product{  
    double cost;  
    ...  
}
```

```
public class VendingMachine{  
    ArrayList<Product>;  
    Product getProductBy...()  
}
```





Перерыв?

Голосуйте в чате



Полиморфизм





Полиморфизм - Задание

```
public class Product{  
    public String toString()  
}
```

```
public class VendingMachine{  
    ArrayList<Product>;  
    Product getProduct(Tea);  
    Product getProduct(Chips);  
    ....  
}
```





Полиморфизм

1. Позволяет подменять реализации объектов. На этом основано тестирование.
1. Обеспечивает расширяемость программы — становится гораздо легче создавать задел на будущее.
1. Позволяет объединять объекты с общим типом или поведением в одну коллекцию или массив и управлять ими единообразно
1. Гибкость при создании новых типов: вы можете выбирать реализацию метода из родителя или переопределить его в потомке.



Домашнее задание



Вопросы?

Вопросы?



Вопросы?





Подведем итоги



Что было
сложного на
семинаре?





Напишите 3 вещи в
комментариях, которым
вы научились сегодня.





Как настроение?





Спасибо за работу!