# Домашняя работа

# Строки/меню

Необходимо реализовать консольную программу, которая бы фильтровала поток текстовой информации подаваемой на вход и на выходе показывала лишь те строчки, которые содержат слово передаваемое программе на вход в качестве аргумента.  
  
Дополнительно:

1. Программа не должна учитывать регистр
2. В аргументах может быть передано не одно слово, а несколько
3. В качестве аргумента может быть задано не конкретное слово, а [регулярное выражение](http://ru.wikipedia.org/wiki/%D0%A0%D0%B5%D0%B3%D1%83%D0%BB%D1%8F%D1%80%D0%BD%D0%BE%D0%B5_%D0%B2%D1%8B%D1%80%D0%B0%D0%B6%D0%B5%D0%BD%D0%B8%D0%B5)

# ООП

Реализовать программу «Банкомат».

Программа дает возможность:

1. Внести деньги на счет
2. Вывести деньги со счета

Дополнительно:

1. Реализовать несколько типов счетов- «Текущий», «Дебетовый», «Кредитный».
2. Реализовать перевод между счетами
3. Реализовать правила связи между счетами-
   1. Снятие с любого счета более чем 30 000 за сеанс запрещено
   2. Наличие кредитного счета с балансом менее минус 20 000 запрещает работу с дебетовым счетом
   3. При пополнении текущего счета более чем на 1 000 000 за операцию- на дебетовый счет автоматически поступает +2 000

Пожелания:

1. Использовать иерархию наследования и другие приемы ООП для устранения дублирования кода
2. Все действия должны быть оформлены текстовыми подсказками на консоли, исключающими неточности
3. Обрабатывать некорректный ввод пользователя

# Массивы

Напишите программу Poker.java, которая должна имитировать раздачу карт для игры в покер. Программа получает число n, задаваемое с консоли пользователем, и раздает карты на n  игроков (по 5 карт каждому) из рассортированной колоды. Разделяйте пять карт, выданных каждому игроку, пустой строкой.

# 4 Коллекции

Реализовать собственный ArrayList или LinkenList.

Реализовать в нем все методы интерфейса Collection и List

# 5 Reflection

Дополнить проект ATM следующим функционалом:

1. Добавить аннотацию @Loggable к классу «Счет», которая включает логгирование всех методов класса
2. Добавить аннотацию @OperationLimit(value= ) для банкомата. Аннотация включает операционный лимит для всех счетов в банкомате
3. Добавить аннотацию @MethodLimit(value= ) для метода счета. Аннотация включает лимит вызовов для метода счета
4. Добавить proxy для класса ATM, который подсчитывает время выполнения метода счета. И если время выше 2 сек, то вызывает метод другого метода.

# 6 Out Clients

Дополнить проект ATM следующим функционалом:

1. Функцией «Сделать дамп в XML-файл»
2. Функцией «Сделать дамп в JSON-файл»
3. Функцией «Сделать восстановление из дампа XML»
4. Функцией «Сделать восстановление из дампа XML»
5. Функцией «Сделать дамп в базу данных»
6. Функцией «Восстановить из базы данных»

# 6 Web Services

Переместить всю работу с ATM в SOAP-вебсервис с функциями:

1. Пополнить счет. Аргументы- название счета, сумма
2. Снять со счета. Аргументы- название счета, сумма

Важно- все аннотации и ограничения из предыдущих пунктов должны быть учтены