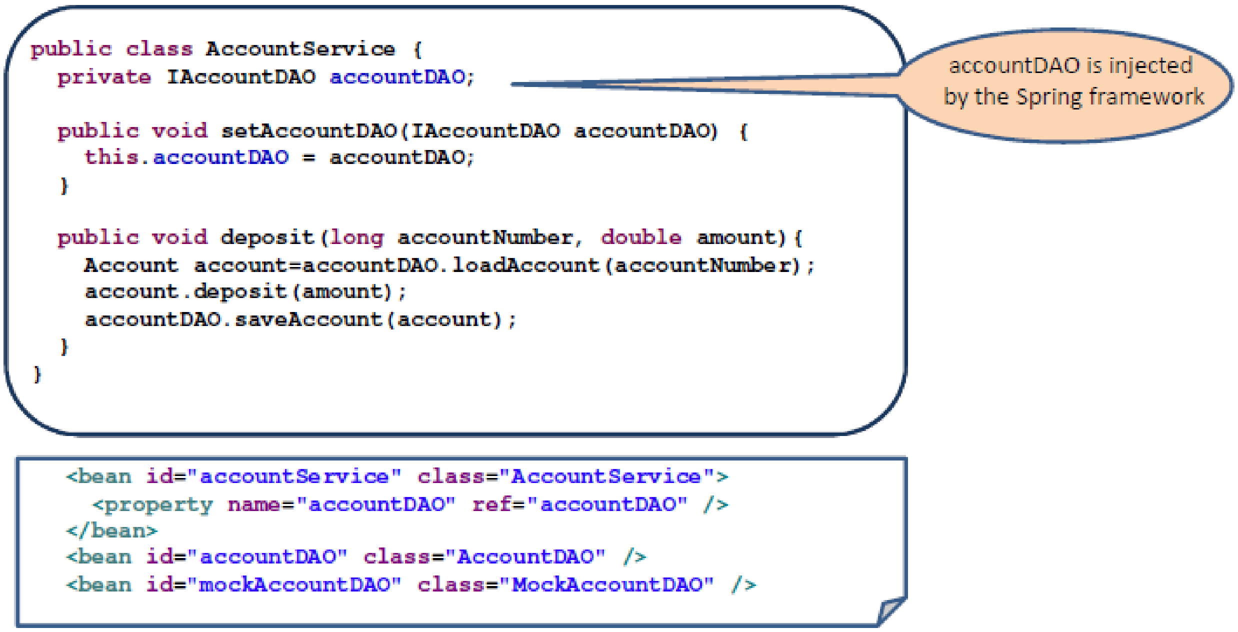
Question 1

a. Flexible way to wire objects together.
b. Flexibility. Easy to change the wiring of objects
c.



```
public class AccountService {
  private IAccountDAO accountDAO;

  public void setAccountDAO(IAccountDAO accountDAO) {
    this.accountDAO = accountDAO;
  }

  public void deposit(long accountNumber, double amount){
    Account account=accountDAO.loadAccount(accountNumber);
    account.deposit(amount);
    accountDAO.saveAccount(account);
  }
}
```

accountDAO is injected by the Spring framework

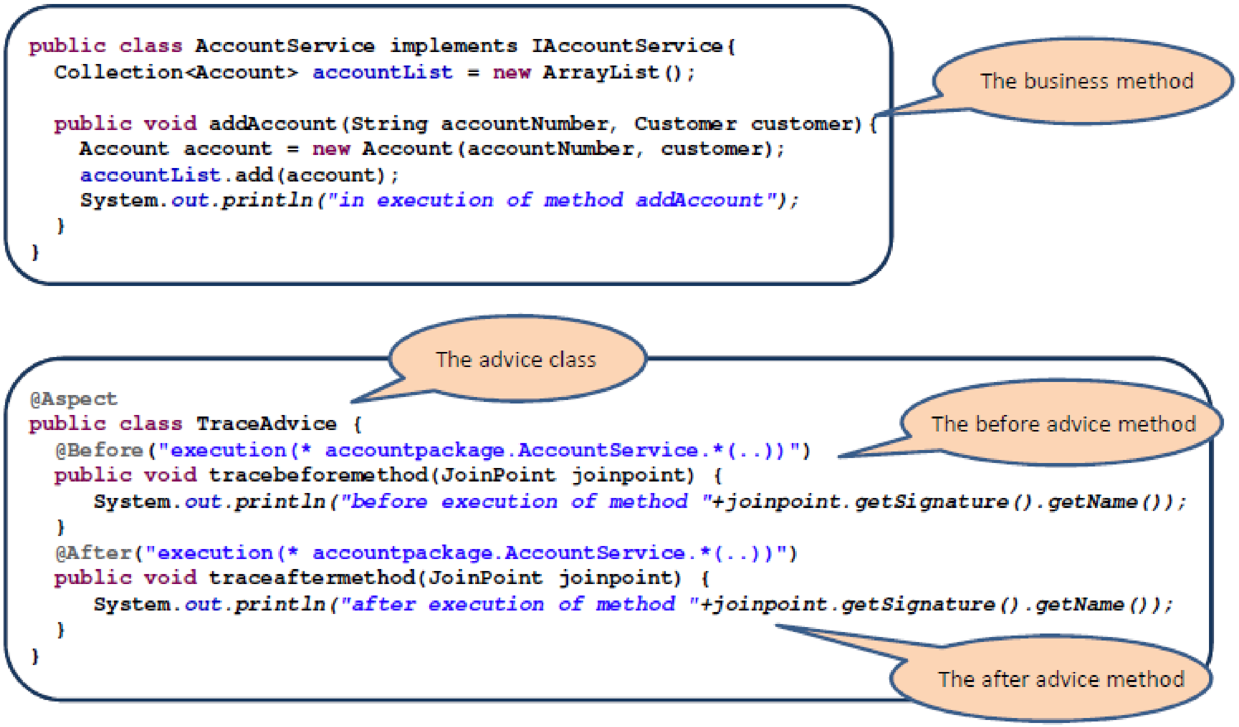
```
<bean id="accountService" class="AccountService">
  <property name="accountDAO" ref="accountDAO" />
</bean>
<bean id="accountDAO" class="AccountDAO" />
<bean id="mockAccountDAO" class="MockAccountDAO" />
```

Question 2

a. Write crosscutting concerns at one place and use them at different places in your application at runtime
b. SOC: Separate functionality at design time, but weave them together at runtime

DRY: Don’t repeat yourself.

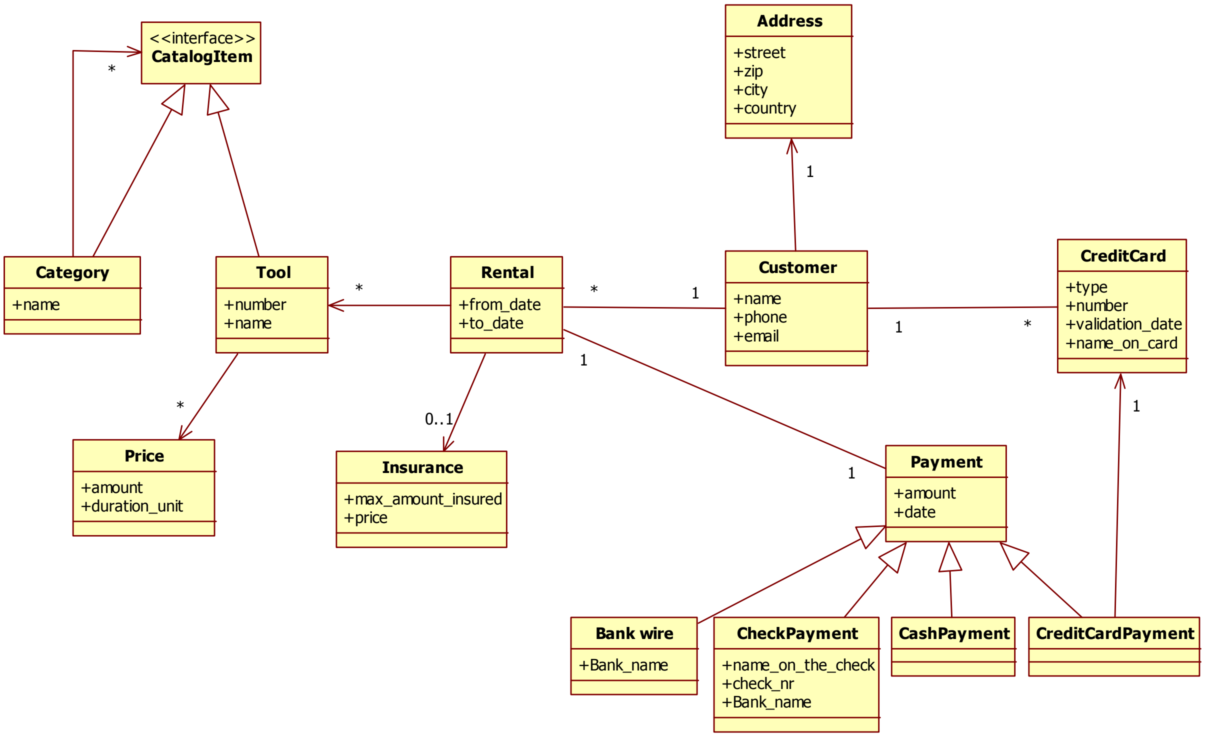
c.



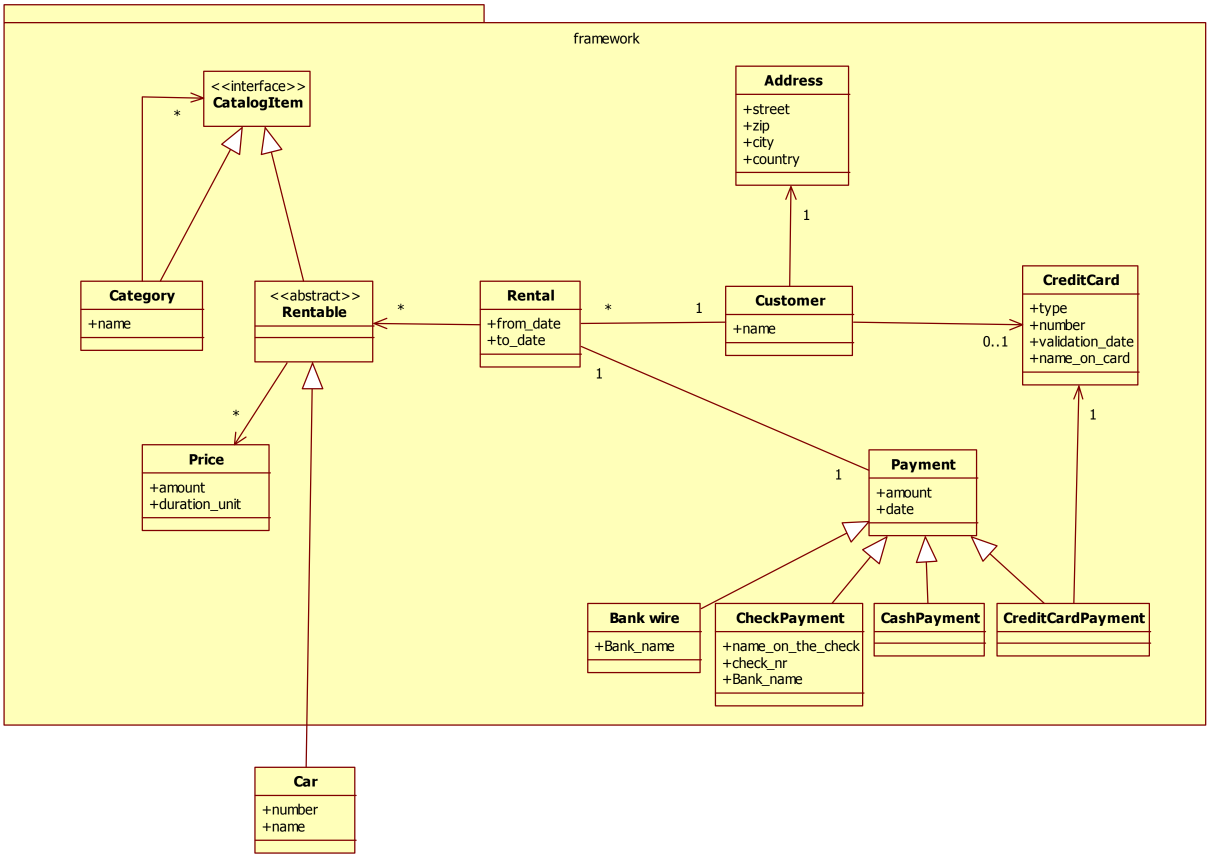
```
public class AccountService implements IAccountService{
  Collection<Account> accountList = new ArrayList();

  public void addAccount(String accountNumber, Customer customer){
    Account account = new Account(accountNumber, customer);
    accountList.add(account);
    System.out.println("in execution of method addAccount");
  }
}
```

```
@Aspect
public class TraceAdvice {
  @Before("execution(* accountpackage.AccountService.*(..))")
  public void tracebeforemethod(JoinPoint joinpoint) {
     System.out.println("before execution of method "+joinpoint.getSignature().getName());
  }
  @After("execution(* accountpackage.AccountService.*(..))")
  public void traceaftermethod(JoinPoint joinpoint) {
     System.out.println("after execution of method "+joinpoint.getSignature().getName());
  }
}
```

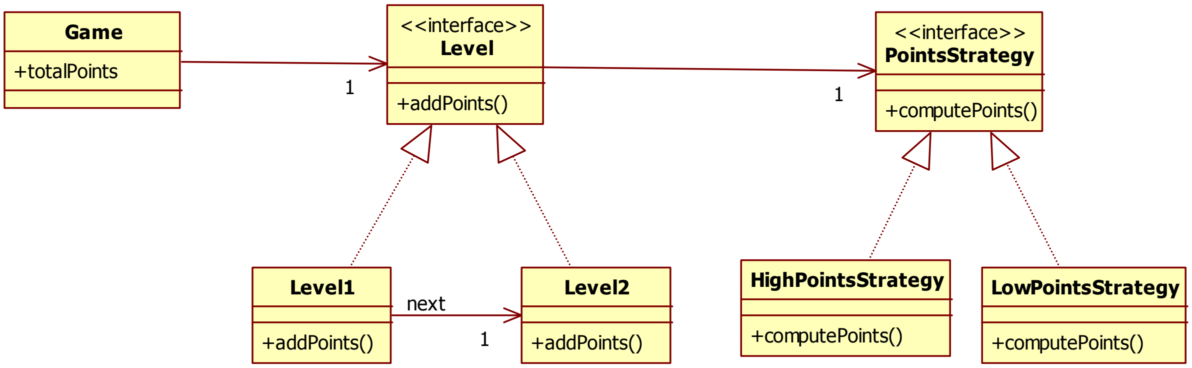
Question 3

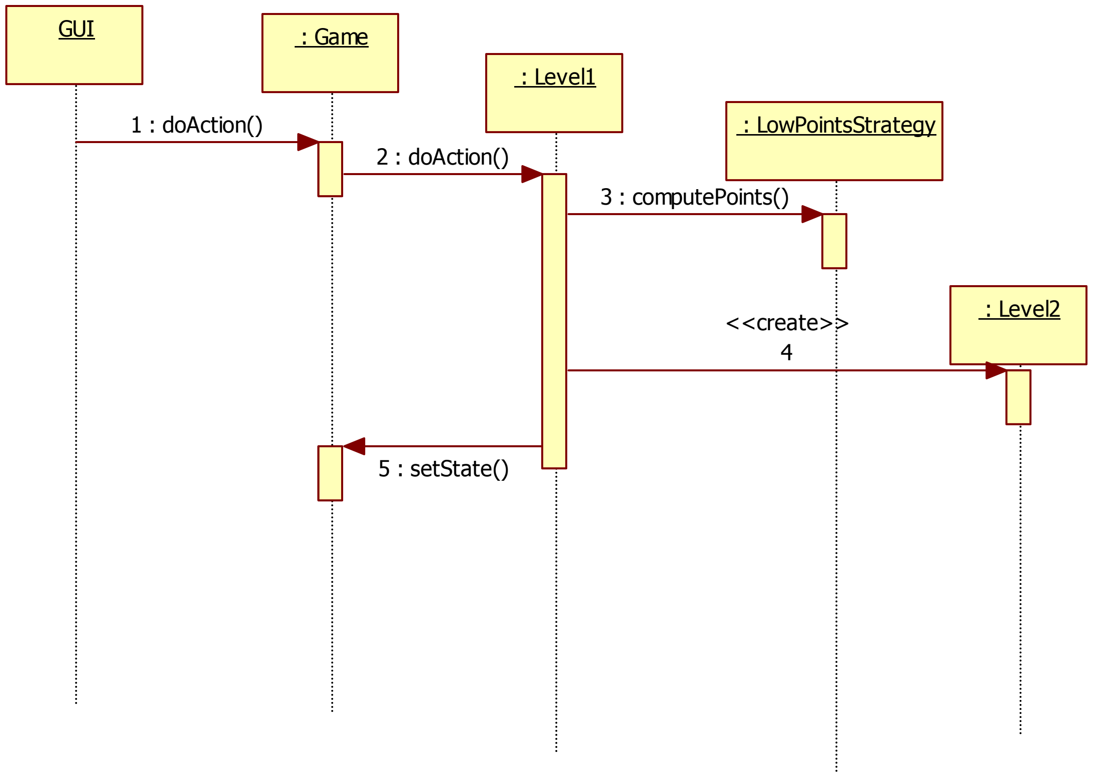

Address
+street
+zip
+city
+country
<<interface>>
CatalogItem
*
1
Category
+name
Tool
+number
+name
*
Rental
+from_date
+to_date
*
1
Customer
+name
+phone
+email
1
*
CreditCard
+type
+number
+validation_date
+name_on_card
1
*
Price
+amount
+duration_unit
0..1
Insurance
+max_amount_insured
+price
1
1
Payment
+amount
+date
Bank wire
+Bank_name
CheckPayment
+name_on_the_check
+check_nr
+Bank_name
CashPayment
CreditCardPayment

Question 4



Question 5




GUI
: Game
: Level1
: LowPointsStrategy
: Level2
1 : doAction()
2 : doAction()
3 : computePoints()
<<create>>
4
5 : setState()