COS 214, Semester Test 1

Matthew Schoeman

U17029377

Question	Points	Score
1	15	5
2	5	4
3	3	2
4	22	5½
5	10	4
6	18	13
Total:	73	

1)

a)

i) Yes

ii) President can be instantiated using the BodyGuard constructor.

iii) Sniper cannot access the private methods inside BodyGuard, therefore creating an undefined reference to those methods. If Sniper was declared a friend class in BodyGuard, then Sniper would have access to BodyGuard's constructor and allowed to be instantiated.

b)

i) It would just output "Employee Salary : " twice with no values.

ii) The reason is perm and temp are not declared as pointers of new objects Permanent and Temporary, respectively. They have not been created as an instance of any class; therefore, the objects are blank. When we expect to see the salary, an empty space is shown because there is not data to show.

iii) Declare the perm and temp variables as Employee pointers of their respective classes.

iv) Polymorphism or Abstract.

2)

a) This would allow modifications to the stored state to happen from the client's side. The purpose of the stored state is to represent the state at the point in time the state was stored. If modifications were made to the stored state, then the purpose of the Memento Pattern is not being fulfilled.

b) Class A is the memento. Class B is the Originator.

3) This is a close attempt, not the Template Method. The base class, Call is not abstract and has no methods for the subclasses to override. A Template Method implementation must have the base class abstract to allow for methods to be overridden in the subclasses, so the subclasses can define the specific steps in the general method used in the base class.

4)

a)

i) Abstract Class

ii) They will be implemented in the Rotary and Pushbutton classes as they are defined as abstract. They are private so no other class can access those methods but the Telephone Class. This helps the security behind having them as private as the two subclasses override

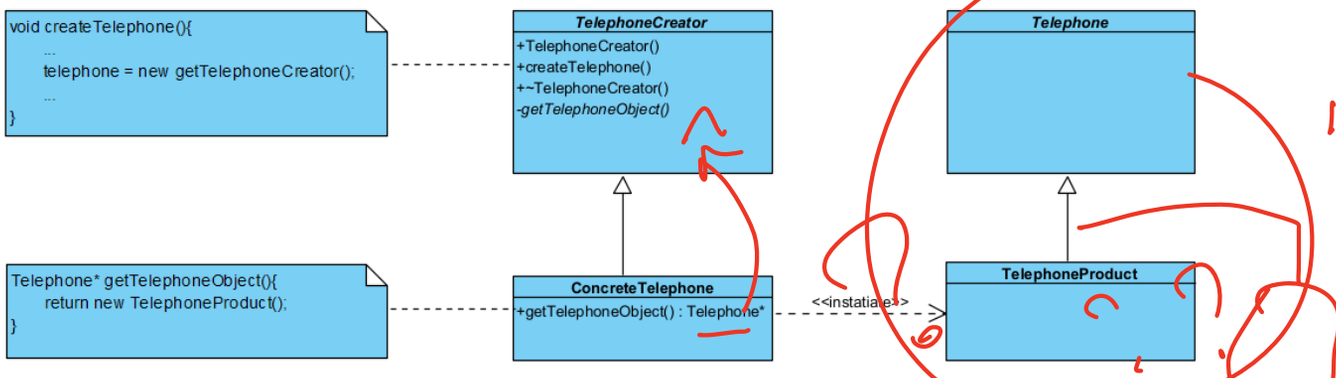
them and can only be used by an instance of the Telephone class when they are needed in useTelephone method.

iii) Template Method.

b)

i) getTelephoneObject() is the factory method. createTelephone() is the operation.

ii)

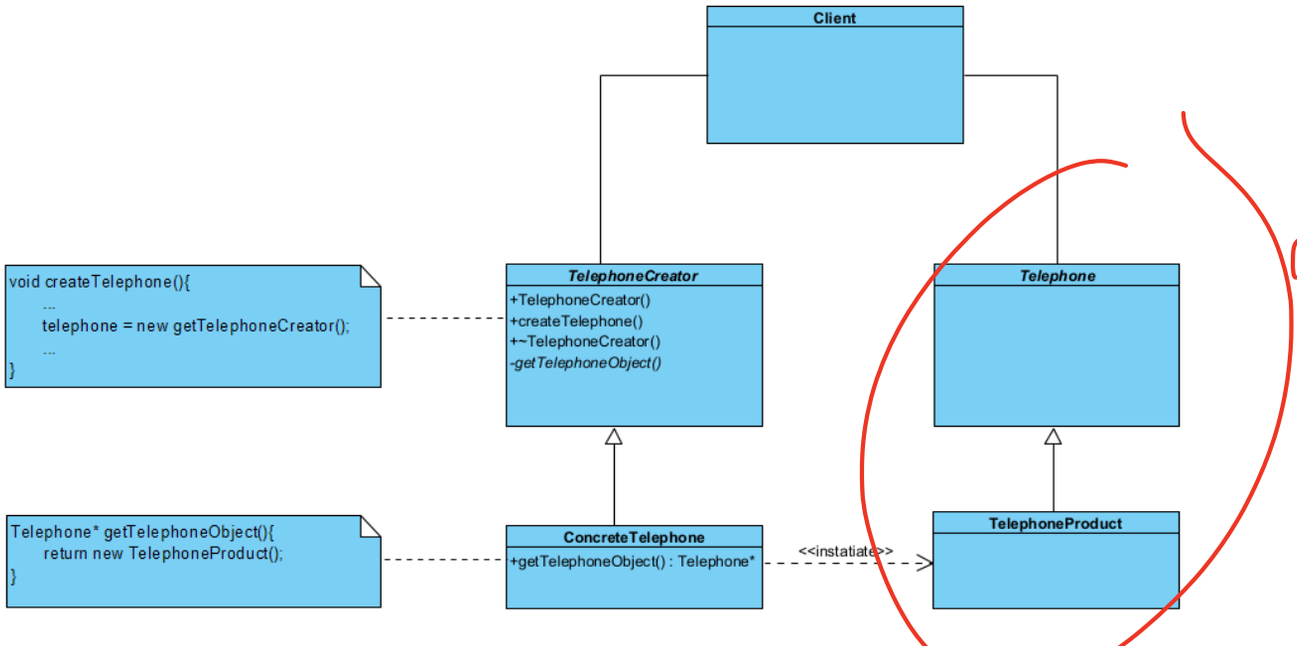


iii) A client class needs to be added with a generic connection to TelephoneCreator class and the Telephone class.

iv)

```
void createTelephone(){
    Telephone* telephone = new getTelephoneCreator();
}
```

v)



c) The Prototype Pattern to make replicas. The Telephone would need two derived classes, one being for rotary and the other for push-button. A Telephone Manager would be needed to create and store multiple Telephone objects in their groups. A cloning function would be

needed to be implemented in the Telephone class which would be overridden in the Telephone subclasses which would return new objects.

5)

a)

i) Telephone Class

ii) Protected: virtual Telephone* clone() = 0;

iii)

```
Telephone* clone(){
    return new Rotary(*this);
}
```

Assumptions made: Rotary is a copy constructor situated in the Rotary class. It accepts a Rotary object which will be used to clone the object. The copy constructor provides a deep copy of the elements, therefore being identical but not sharing the same allocated memory.

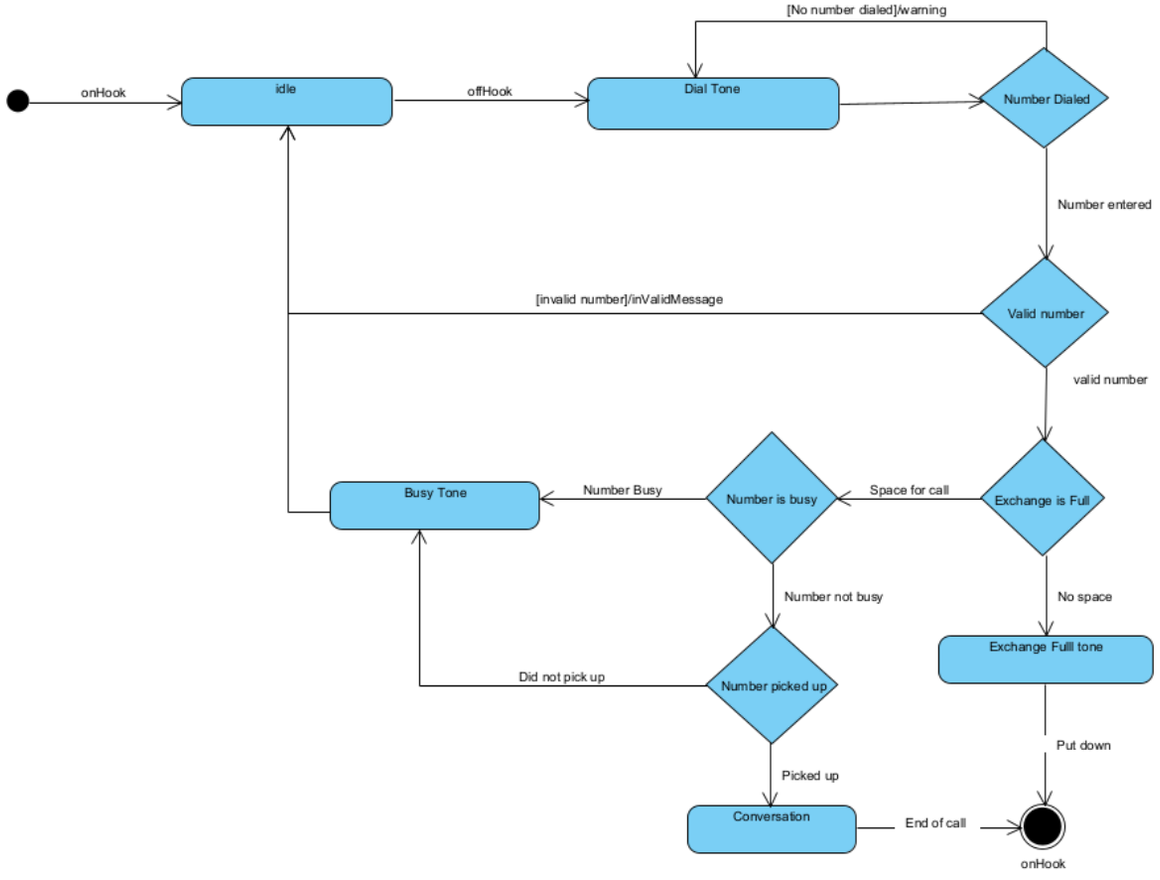
b)

i) Strategy Pattern

ii) The strategy participant exists as the Telephone Manager. The Telephone class maps to the Telephone Manager.

6)

a)

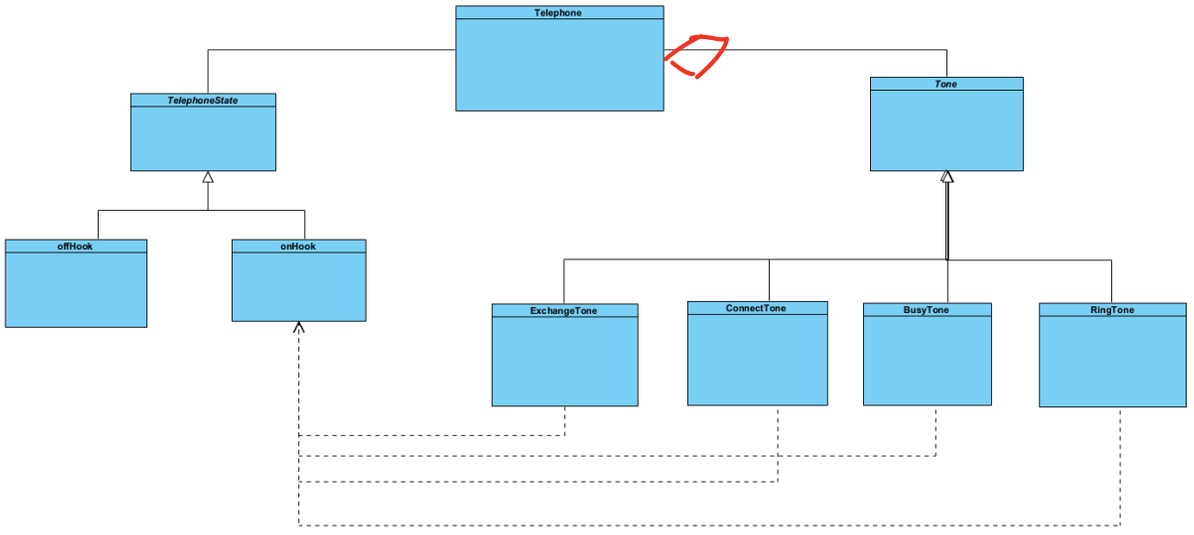


b)

i) The State Pattern. Using this pattern would allow the functionality of the system to alter depending on the type of current state the system is in, and allow what state should be put in next.

2

ii)



4