CS 542 Design Pattern and Object-Oriented Analysis

Lab 6

Student Name	Student CSUSM ID	Contribution percentage
1 Dhruval shah	200361439	33%
2 Niki Patel	200360048	33%
3 Yifan Wu	200379249	33%
4		
5		

Grading Rubrics (for instructor only):

Criteria	1. Beginning	2. Developing	3. Proficient	4. Exemplary
Program: functionality *correctness*	0-9	10-14	15-19	20
Program: functionality *Behavior Testing*	0-9	10-14	15-19	20
Program: quality -> *Readability*	0-9	10-14	15-19	20
Program: quality -> *Modularity*	0-9	10-14	15-19	20
Program: quality -> *Simplicity*	0-9	10-14	15-19	20
Total Grade (100)				

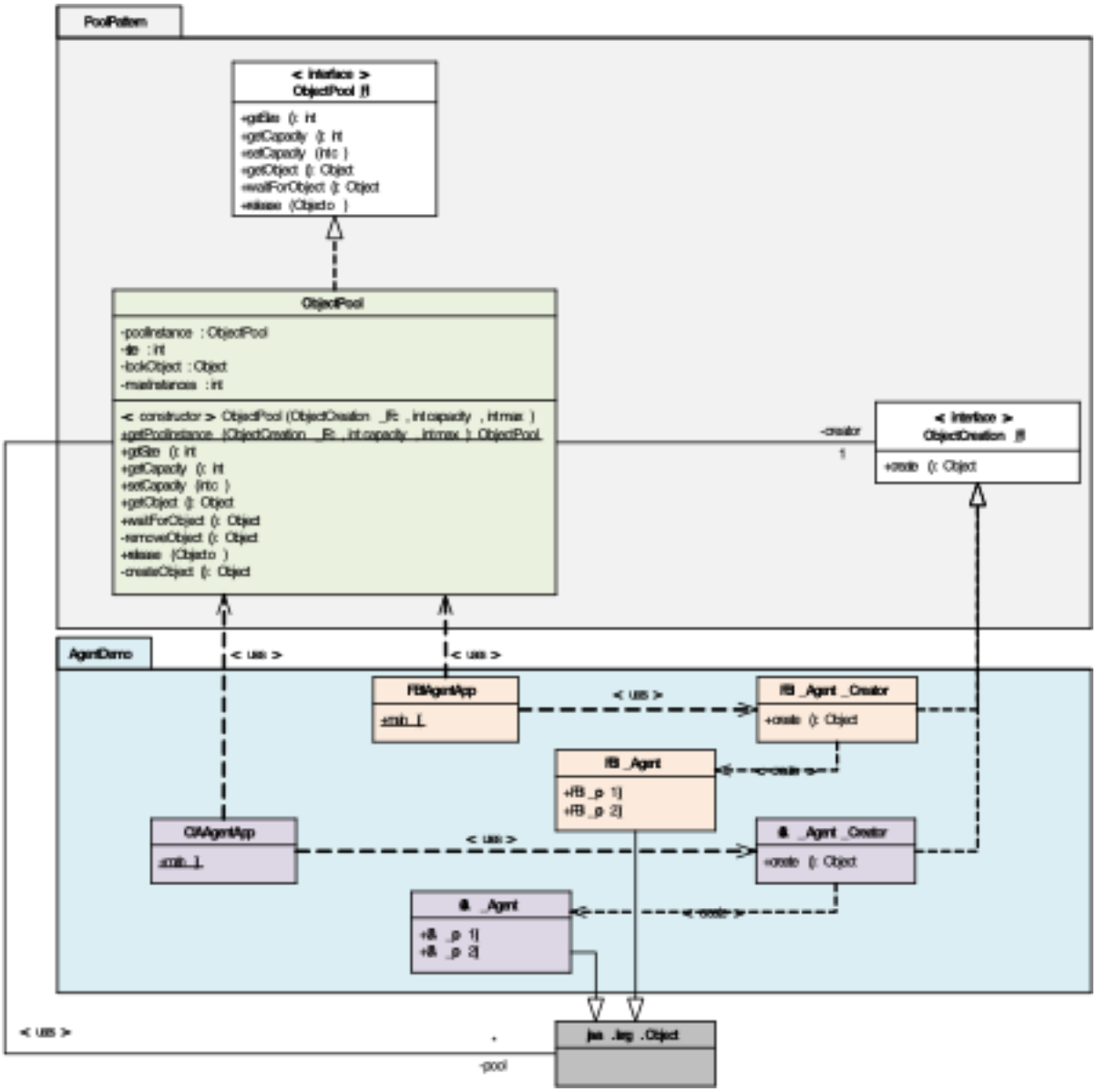
Problems:

Given the following design, implement it in Java. Note:

1. You may add more attributes or operations to a class if necessary. Specifically, you may use meaningful operations for FBI_Agent and CIA_Agent classes.

2. Read textbook to see some example code snippets (pp. 170—174).

3. Your testing code (FBIAgentApp or CIAAgentAPP) should demonstrate how limited number of agents are requested to process tasks (the number of tasks are greater than the number of agents)



Solution:

- First, remember to zip the src folder of your project and submit the zip file to the ungraded assignment named "Lab6CodeSubmission". One submission from each team.

- Paste a screenshot of a run of your program here.

- Also paste all you source code here.

- Save this report in PDF, then **each student** needs to submit the pdf report to the graded assignment named "Lab6ReportSubmission".

- Save this report in PDF, then **each student** needs to submit the pdf report to the graded assignment named “Lab6ReportSubmission”.

Search Results | Output - 542Lab6 (run)

```
run:
Testing FBI Agent pool
This is agent A, work on task 0
This is agent B, work on task 1
This is agent C, work on task 2
This is agent D, work on task 3
This is agent E, work on task 4
This is agent A, work on task 5
This is agent B, work on task 6
This is agent C, work on task 7
This is agent D, work on task 8
This is agent E, work on task 9
BUILD SUCCESSFUL (total time: 2 seconds)
```

Search Results | Output - 542Lab6 (run)

```
run:
Testing CIA Agent pool
This is agent a, work on task 0
This is agent b, work on task 1
This is agent c, work on task 2
This is agent d, work on task 3
This is agent e, work on task 4
This is agent a, work on task 5
This is agent b, work on task 6
This is agent c, work on task 7
This is agent d, work on task 8
This is agent e, work on task 9
BUILD SUCCESSFUL (total time: 2 seconds)
```

package PoolPattern

ObjectPool_IF.java

```
package PoolPattern;
public interface ObjectPool_IF {

    public int getSize();
    public int getCapacity();

    public void setCapacity(int c);
    public Object getObject();
    public Object waitForObject() throws InterruptedException;
    public void release(Object o);
}
```

ObjectCreation_IF.java

```
package PoolPattern;
public interface ObjectCreation_IF {
    public Object create();
}
```

package PoolPattern

ObjectPool.java

```
package PoolPattern;
public class ObjectPool implements ObjectPool_IF {

    private static ObjectPool poolInstance;
    private final Object lockObject;
    private int size;
    private int maxInstances;
    private int instanceCount;
    private ObjectCreation_IF c;
    private Object[] pool;

    private ObjectPool(ObjectCreation_IF c, int capacity, int max) {
        this.c = c;
        this.size = 0;
        this.instanceCount = 0;
        this.maxInstances = max;
        this.lockObject = new Object();
        pool = new Object[capacity];
    }

    public synchronized static ObjectPool
getPoolInstance(ObjectCreation_IF c, int capacity, int max)
    {
        poolInstance = new ObjectPool(c, capacity, max);
        return poolInstance;
    }
    @Override
    public int getSize() {
        return this.size;
    }
    @Override
    public int getCapacity() {
        return pool.length;
    }
    public int getInstanceCount() {
        return this.instanceCount;
    }
    public int getMaxInstances() {
        return maxInstances;
    }
    @Override
    public void setCapacity(int c) {
        if (c <= 0) {
            String msg = "Capacity must be greater than zero";
            throw new IllegalArgumentException(msg);
        }
        synchronized (lockObject) {
            Object[] newPool = new Object[c];
            System.arraycopy(pool, 0, newPool, 0, c);
            pool = newPool;
        }
    }
    @Override
    public Object getObject() {
        synchronized (lockObject) {
            if (size > 0) {
                return removeObject();
            }
            else if (getInstanceCount() < getMaxInstances()) {
                return createObject();
            }
            else {
                return null;
            }
        }
    }
    @Override
    public Object waitForObject()
    throws InterruptedException {
        synchronized (lockObject) {
            if (getInstanceCount() < getMaxInstances()) {
                return createObject();
            }
            else if (size > 0) {
                return removeObject();
            }
            else {
                do {
                    wait();
                }
                while (size <= 0);
                return removeObject();
            }
        }
    }
    private Object removeObject() {
        size--;
        return pool[size];
    }
    @Override
    public void release(Object o) {
        if (o == null) {
            throw new NullPointerException();
        }
        synchronized (lockObject) {
            if (getSize() < getCapacity()) {
                pool[size] = o;
                size++;
                lockObject.notify();
            }
        }
    }
    private Object createObject() {
        Object newObject = c.create();
        instanceCount++;
        return newObject;
    }
}
```

AgentDemo

FBIAgentApp.java

```
package AgentDemo;
import PoolPattern.ObjectPool;
import java.util.ArrayList;
public class FBIAgentApp {
    public static void main(String arg[]) throws InterruptedException {
        FBI_Agent_Creator creator = new FBI_Agent_Creator();
        ObjectPool CIA = ObjectPool.getPoolInstance(creator, 5, 5);
        ArrayList<Task> list = new ArrayList();
        for (int i = 0; i < 10; i++) {
        Task t = new Task(i);
        list.add(t);
    }
    System.out.println("Testing FBI Agent pool");
    int index = 0;
    Task t = list.get(index);
    while (t != null && index < list.size()) {
        try {
            FBI_Agent agent = (FBI_Agent) CIA.waitForObject();
            t = list.get(index);
            index++;
            if (t != null) {
                agent.setTask(t);
                agent.run();
                Thread t1 = new Thread(() -> {
                try {
                    Thread.sleep(400);
                    CIA.release(agent);
                } catch (InterruptedException ex) {}
            });
            t1.start();
            }
            } catch (InterruptedException ex) {}
        }
    }
}
```

Task.java

```
package AgentDemo;
import PoolPattern.ObjectPool;
import static java.lang.Thread.sleep;
public class Task {
    private int id;
    public Task(int id) {
        this.id = id;
    }
    public void setTaskID(int id) {
        this.id = id;
    }
    public int getID() {
        return id;
    }
}
```

ObjectCreation_IF.java

```
package Agpackage AgentDemo;

import PoolPattern.ObjectPool;
import java.util.ArrayList;
```

```
package AgentDemo;

import PoolPattern.ObjectCreation_IF;
public class FBI_Agent_Creator implements ObjectCreation_IF {
    private String[] names = {"A", "B", "C", "D", "E"};
    private int index;
    public Object create(){
        FBI_Agent agent = new FBI_Agent(names[index++]);
        return agent;
    }
}
```

FBI_Agent.java

```
package AgentDemo;
import static java.lang.Thread.sleep;
public class FBI_Agent extends Object {
    private boolean working;
    int i;
    String name;
    private Task t;
    public FBI_Agent(String name) {
        this.name = name;
    }
    public void setTask(Task t) {
        this.t = t;
    }
    public Task getTask() {
        return t;
    }
    public void run() {
    try {
        sleep(100);
        System.out.println("This is agent " + name + ",
                        work on task " + getTask().getID());
    } catch (InterruptedException ex) {}
    }
    public synchronized void start() {
        this.working = true;
    }
    public synchronized void stop() {
        this.working = false;
    }
}
```

CIA_Agent_Creator.java

```
package AgentDemo;
import PoolPattern.ObjectCreation_IF;
public class CIA_Agent_Creator implements
ObjectCreation_IF{
    private String[] names = {"a", "b", "c", "d", "e"};
    private int index;
    public Object create(){
        CIA_Agent agent = new CIA_Agent(names[index++]);
        return agent;
    }
}
```

CIAAgentApp.java

```
package AgentDemo;
import PoolPattern.ObjectPool;
import java.util.ArrayList;
public class CIAAgentApp {

public static void main(String arg[]) throws
InterruptedException {
    CIA_Agent_Creator creator = new CIA_Agent_Creator();
    ObjectPool CIA = ObjectPool.getPoolInstance(creator, 5, 5);
    ArrayList<Task> list = new ArrayList();
    for (int i = 0; i < 10; i++) {
        Task t = new Task(i);
        list.add(t);
    }
    System.out.println("Testing CIA Agent pool");

    int index = 0;
    Task t = list.get(index);
    while (t != null && index < list.size()) {
        try {
            CIA_Agent agent =
                    (CIA_Agent) CIA.waitForObject();
            t = list.get(index);
            index++;
            if (t != null) {
                agent.setTask(t);
                agent.run();
                Thread t1 = new Thread(() -> {
                try {
                    Thread.sleep(400);
                    CIA.release(agent);
                } catch (InterruptedException ex) {}
            });
            t1.start();
        }
        } catch (InterruptedException ex) {}
    }
    }
}
```

CIA_Agent.java

```
package AgentDemo;
import static java.lang.Thread.sleep;
public class CIA_Agent extends Object {
    private boolean working;
    int i;
    String name;
    private Task t;
    public CIA_Agent(String name) {
        this.name = name;
    }
    public void setTask(Task t) {
        this.t = t;
    }
    public Task getTask() {
        return t;
    }
    public void run() {
        try {
            sleep(100);
            System.out.println("This is agent " + name + ", work on task " + getTask().getID());
        } catch (InterruptedException ex) {}
    }
    public synchronized void start() {
        this.working = true;
    }
    public synchronized void stop() {
        this.working = false;
    }
}
```