Skeleton_Ass2

public class Board

private final Road[] roads;
private final Knight[] knights;
private final Settlement[] settlements;
private final City[] cities;
private int round;
private int[] scoresRecorder；

public Board(Road[] roads, Knight[] knights, Settlement[] settlements, City[] cities){}

public boolean tryToBuild(BuildableStructures whatWeWantToBuild, Resources[] resourcesWeHave){}

public void build(BuildableStructures whatWeWantToBuild) {}

public boolean whetherShouldFinish(){}

public int calculateFinalScores(){}

public Resources[] firstRoll(){}

public Resources[] rollAgain(int[] positionOfResourcesWantToReplace() public Resources[] tradeByGold(Resources resourceWanted){}

public Resources[] tradeWithKnight(Knight knight, Resources resourcePaid){}

public class BuildableStructures

protected int x;
protected int scores;
protected boolean whetherHaveBuilt=false;
protected Resources[] demandOfResources;

public int getX(){}

public int getScores(){}

public boolean isWhetherBuild(){}

public void setWhetherBuild(boolean whetherBuild){}

public Resources[] getDemandOfResources(){}

public class Road extends BuildableStructures

private int y;

public Road(int x, int y, int scores, Resources[] demandOfResources){}

public int getY(){}

public class Knight extends BuildableStructures

private boolean whetherHaveSwapped=false;

public Knight(int x, int scores, Resources[] demandOfResources){}

public boolean isWhetherHaveSwapped(){}

public void setWhetherHaveSwapped(boolean whetherHaveSwapped){}

public class Settlement extends BuildableStructures

public Settlement(int x, int scores, Resources[] demandOfResources){}

piblic class City extends BuildableStructures

public City(int x, int scores, Resources[] demandOfResources){}

Public enum Resources

Grain
Wool
Timber
Brick
Gold
Ore