Algorithms 3

Project – Strategy Game

Team 2: Nour Eldin CHARIFH, Tiago Goncalves Da Silva, Yinghan LING

Town Center (Tiago Goncalves Da Silva):

Food resource:

- Food is an integer variable
- We have a method for gathering food (for workers) which is private
- We have a method build Workers which checks if we have 25 food unit, to create one worker

Build Workers:

- We have an Array list of objects with [int: **ID**, Boolean: **availably**, String: **task**], referring to each element (worker) uniquely
- Garnering food method can be triggered from the Town Center part
- Gathering wood method can be triggered from the workshop part

Unit limit:

- A Boolean method to control the maximum number of units that can exist (the condition of upgrade)
- Another method to check the number of units taken by any players.

Upgrade Town Center:

- We have upgrade method which will turn a Boolean global variable from false to true.
- Town center can run upgrade method, but should get access into wood and gold variables in the other parts.

Status:

We have status method to display info about unit limit, current food, available workers, active workers.

Workshop (Yinghan LING):

Wood resource:

- Wood should be a class with three int properties where we add the gathered value properly.

Gather wood:

- We have the gather wood method which takes workers as a parameter
- Workers should be retrieved from Town Center
- The equation for gathering wood is 1* #workers, and the type of wood is gathered randomly between the three types.

Boost gathering:

- We have “boost gathering method which takes food as a parameter and a specific worker (by the unique id) which should be working in wood gathering.

Build houses:

- We have a method build house which reduce the wood value by 40, and then for each house we build, our limit of units (which is in Town Center) increases by 5.

Status:

We have status method to display info about current wood, active workers gathering wood (from Town Center), houses.

Barracks (Nour Eldin CHARIFH):

Build Archers:

- Barracks is a class which has a method to build archers.
- We’re going to have a variable called archers (initially 0) which accumulate all archers from all barracks.
- We have the build archer method which will request access into the other parts (Town Center & Workshop) in order to use variables and arrays (food, wood, workers) as parameters.
- We have a method which returns what we retrieved if the other resources are not available. We use it after if condition to check if all resources are available.
- NOTE: build archer method will consume (decrease) food and wood values but workers value will be back after building the archer.

Gold resource:

- Gold should be a variable of the barracks class which increases (when invoking “plunder” method) by (5,25, or 50) depending to its type

Plunder treasures:

- We have “plunder” method which runs automatically (randomly every 5 to 8 seconds) and takes archers as a parameter.

- In the body of plunder method, we increase Monuments and/or gold values, applying probability of (25% 75%) respectively.
- Monuments is a variable which will be increased probably after running the plunder method.

NOTE: an archer can't find monuments before one half upgrade.

- Within the method body we apply probability of losing the archer (50% before upgrade, 25% after upgrade)
- A method to inform the Town Center about the unit limit after each plunder method run.

<u>Upgrade archers:</u>

- A method in barracks class to upgrade its archers which takes an archer as a parameter and reduce the risk of dying from 50% to 25% (can be applied once for each archer)
- This method will require access into (Town Center and Workshop) to retrieve data (100 units of wood and 50 units of food)

<u>Status:</u>

We have status method to display info about Gold, active archers, Monuments.

==

<u>Concurrency:</u>

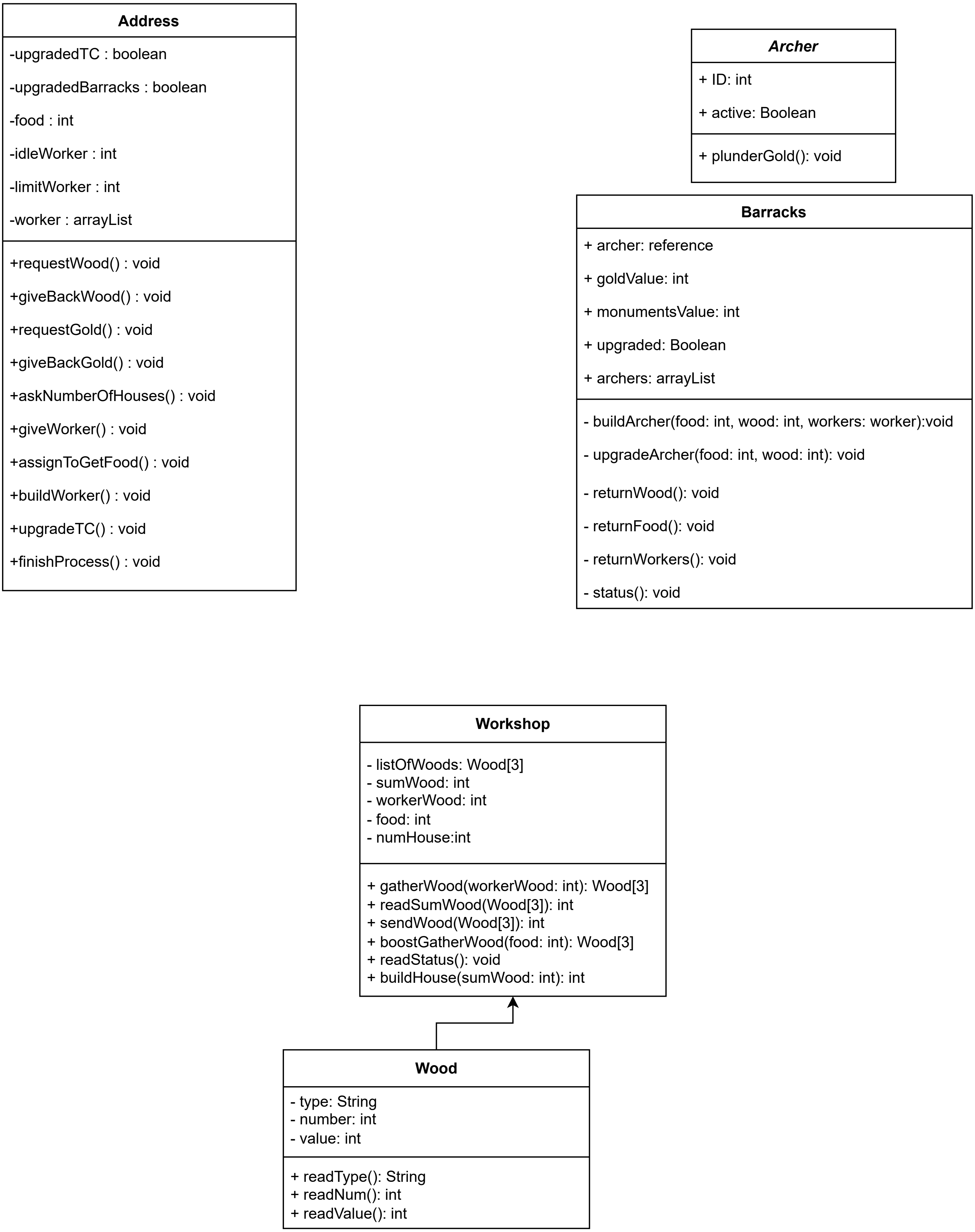
Town Center: In the town center we have the workers which we are going to use Collections<Workers> ArrayList and the class Workers consist of "public record Worker (String task, …)"

Workshop && Barracks: In the workshop and barracks we have the wood/Gold which we are going to use Collections<Wood>/Collection<Gold> ArrayList and the class Wood/Gold consist of "public record Gold (String kindOfGold, int amount)"/

"Public record Wood (String kindOfWood, int amount)"

<u>Communication:</u>

The three of us decided to use java, so we are also going to use javelin to connect our code to one chosen port. We will use the HTTP operation (GET, POST, PUT, DELETE) for the methods that request any concurrency of the sub-parts.



Address

-upgradedTC : boolean

-upgradedBarracks : boolean

-food : int

-idleWorker : int

-limitWorker : int

-worker : arrayList

+requestWood() : void

+giveBackWood() : void

+requestGold() : void

+giveBackGold() : void

+askNumberOfHouses() : void

+giveWorker() : void

+assignToGetFood() : void

+buildWorker() : void

+upgradeTC() : void

+finishProcess() : void

Archer

+ ID: int

+ active: Boolean

+ plunderGold(): void

Barracks

+ archer: reference

+ goldValue: int

+ monumentsValue: int

+ upgraded: Boolean

+ archers: arrayList

- buildArcher(food: int, wood: int, workers: worker):void

- upgradeArcher(food: int, wood: int): void

- returnWood(): void

- returnFood(): void

- returnWorkers(): void

- status(): void

Workshop

- listOfWoods: Wood[3]
- sumWood: int
- workerWood: int
- food: int
- numHouse:int

+ gatherWood(workerWood: int): Wood[3]
+ readSumWood(Wood[3]): int
+ sendWood(Wood[3]): int
+ boostGatherWood(food: int): Wood[3]
+ readStatus(): void
+ buildHouse(sumWood: int): int

Wood

- type: String
- number: int
- value: int

+ readType(): String
+ readNum(): int
+ readValue(): int