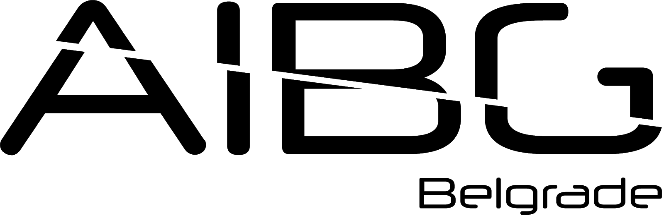
Version 1.0

06/12/2019

ASSIGNMENT

Assignment description and instructions for solving





 AIBG 2.0

BEST BEOGRAD

KARNEGIJEVA 4

**Story 2.0**

Aka **W**hy robots used time machine

Our story begins in the future. The year is 3019. Robots have completely replaced the human workforce. Equipped with the latest technologies, they feature the best AI agents, created for the most diverse purposes. There are robots that play, those who tidy up, cook, and even those who build and create.

But something is missing, everything they build collapses, dissipates and disappears. Despite all the efforts to acquire and absorb the skills of the old masters, every attempt is followed by failure. They are unable to build grandiose works of ancient times.

Famous scientists from all countries are coming together to solve this problem.

The only solution: send robots through time, into the past, into the Middle Ages, when they are magnificent fortresses built, to learn and apply ancient skills.

At first, everything went according to plan, the robots were unnoticed collecting data and they acquired very fast the necessary knowledge.

But scientists have made one error that was an oversight. The robots were sent to the core of the war between two large dynasties. Depending on whose territory the robot came to, he was collecting that dynasty’s data. He learned their medals, titles, values.

Soon, all the robots picked sides. Instead of collecting data passively, they got involved in the battle of the two kings. They use all the necessary resources: stone, wood, metal and knowledge and skills their acquired to create large fortresses equipped to fight and bring victory to their king. The great dynasty war turned into a war for the future.

Who will have a better strategy for destroying the enemy?

Who will win?

# Rules of the game

## Game description

AI Battleground 2019 is a TBS (Turn-Based Strategy) game in which two artificial intelligence they take on the role and powers of the Avatars to fight each other. The moves take place intermittently, and the first move always has Avatar Number 1.

Avatars can move around the map, gather resources to build weapons-generating fortresses.  Using weapons, they can attack the opponent and his structures, as well as protect against the attacks of the opponent. The goal of the game is to reduce life points opponent's Avatar at 0 through a number of moves and have the highest score. Each game consists of 5 rounds and the winner is the Avatar who first manages to defeat an opponent 5 times or had the highest score.

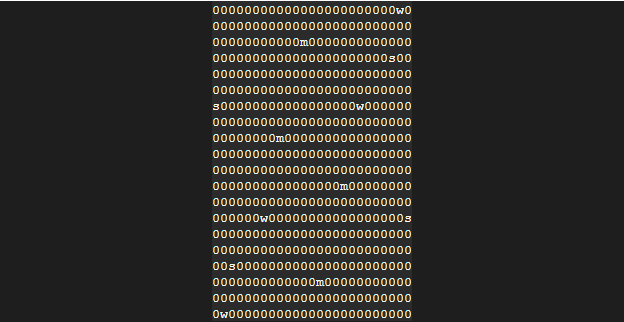
The task is to create a computer program that will play the game as one of the Avatars that is

as intelligent as possible using server-side requests and server responses in JSON format.

## Maps

Maps are randomly generated. Example of the map:

0000000000000000000000000000

00000000000m0000000000

Each character represents one map field:

● 0 - indicates that this field is blank

**●** s - indicates that there is a stone shop in that field

**●** m - indicates that there is a metal shop in that field

● w - indicates that there is a woodshop in that field

Avatar number one is created in the upper left corner of the map and Avatar number two is created in the lower right corner.

The size of the map is constant: 25 x 20. The map is not symmetrical, but all shops are always on the same distance from both players.

## Avatars

Avatars have the following attributes:

* **Health Points (HP)** starts from the maximum value of 100. If HP drops to 0 Avatar loses the round. The positions of both Avatars then reset to the starting position.
* **Lives** - at the beginning, avatars have 5 lives. If lives go down to level 0, that Avatar loses the game.
* **Points (score)** that are collected by construction or demolition of buildings and killing the opponent.
* **Stupid move (stupid moves):** a stupid move is a form of illegal activity, it is an action that cannot be executed. When an avatar makes 10 stupid moves, his HP is decreased by 10.

## Actions

Each Avatar must create a single action command per move. Possible actions are:

* **Moving up, down, left and right for each field** .
  + If the avatar wants to move to field where there is an opponent, shop, or object such as building or building under construction, movement is not possible.
  + If the field is out of borders of the map, moving is not possible.
  + Actions for movement: **w** - up, **s** - down, **a** - left, **d** - right.
* **Take resources from the store**
  + There are resource shops located in certain map fields. Any store can contain only one type of resource: wood, stone or metal. A maximum of 10 resources can be found in one store. When the store is empty (store resources are at 0), for 3 x 2 moves will again be filled to maximum capacity (when both players play 3 strokes each). A player can only take a resource from the store if it is located below, above, to the left or right of it. A player can have a maximum of 5 resources.
  + It is not allowed to take the resource when the store is empty or when the player has 5 resources.
  + Actions for taking resources are: **trw** - taking resources from the store above, **trs** - taking resources from the store below, **tra** - taking resources from the store to the left, **trd** - taking resources from the store to the right.
* **Dropping resources**
  + The player can remove in one action one resource that he currently carries with him. It's not allowed to drain a resource if he does not own.
  + Actions for releasing resources are: **lw** - releasing wood, **ls** - drain stone **lm** -  the discharge of metals.
* **Construction of: houses, forts, fortifications real swords, arrows and shields**
  + Shops are not a facility. The player can first build a house, then he can upgrade home to a fortress, and then fortress into a fortress that makes swords, arrows, or shields depending on of funds collected. The resources needed to build these facilities are:
    - House - 2 x wood, 1 x stone
    - Fortification - 3 x stone, 2 x wood (+ pre-built house)
    - Shield fortification - 2 x metal (+ pre-built fort)
    - Sword fortification - 3 x metal, 1 x wood (+ pre-built fort)
    - Lightning Fort - 4 x metal, 1 x wood (+ pre-built fort)
      * + You can start construction in the field above, below, left and right of itself. It is not possible to upgrade the building while it was still under construction (flag *buildInProcess* ) or build a building for which he doesn’t have enough resources. Construction for each of these objects takes a certain number of moves and for each facility that he built, the player is awarded points *(score)* :

House - 2 x 2 moves (after both players play 2 moves each then house is built)  *+ 15 score*

Fortress - 3 x 2 moves (after both players have played 3 moves each then a fortress is constructed) *+ 10 score*

Shield Fortress - 3 x 2 moves (after both players play 3 moves each) fortress is built to shield) *+ 5 scores*

Sword Fortress - 4 x 2 moves (after both players play 4 moves each) fortress is built for swords) *+ 5 scores*

Arrow Fortress - 5 x 2 moves (after both players have played 5 moves each) fortress is built for arrow) *+ 5 scores*

* + Actions for construction of facilities are:
    - building a house: **bhw** - up, **bhs** - down **bha** - left , **bhd** - right.
    - construction of fortress: **bfw** - up, **bfs** - down **bfa** - left, **bfd** - right.
    - construction of shields fortress: **bshfw** - up, **bshfs** - down **bshfa** - left, **bshfd** - right.
    - construction of sword fortress: **bsfw** - up, **bsfs** - down **bsfa** - left, **bsfd** - right.
    - construction of arrow fortress: **bafw** - up, **bafs** - down **bafa** - left, **bafd** - right.

* **Taking weapons from the fortress which produces that weapon** .
  + Every fortress for making weapons is empty at the beginning. For making every weapon it takes a while. A player can only take a weapon from his fortress, and that if there is at least one made piece of a weapon in the fortress. The player can carry with him a maximum two weapons, if he already has two weapons then he is not allowed to take with him third
  + To make one weapon in the fortress requires: 7 x 2 shield moves, 10 x 2 moves for sword, 10 x 2 strokes for a pack of arrows. The maximum number of weapons in a fortress is 5. Weapons are regenerated when there are less than 5 weapons in the fortress. The player can take from fortress only if it is located above, below, left and right of it.
  + Actions for taking the weapons are: **tww** - up, **tws** - down **twa** - left **twd** - right.

* **The attack on opponents/facility with a sword** .
  + With one sword, a player can attack 5 times. Each attack takes down to an opponent/object 10HP. The player can only attack objects or opponent. All objects are allowed to be attacked, but only by demolishing an opponents object does the player get points. Attacks on stores or on blank fields are not allowed and wastes one swing with a sword. Sword attack can be used on the opponent / object that is one field away (up, down, right, or left). If a player has two swords, he will first use the sword to attack with less swings.
  + Actions for sword attack are: **saw** - up, **sas** - down **saa** - left, **sad** - right.

* **The attack on opponents/object with an arrow**.
  + A single arrow attack takes down an opponent/object 5HP, there are 5 in one arrow pack pieces. An arrow can attack an opponent/object at a maximum distance of 3 fields: (the player can attack the highlighted fields in the image). All objects are allowed to attack, but only by demolishing an opponent's object does the player get points. Attack on stores or on blank fields are not allowed and consumes one arrow. If both weapons are arrows, the attack is selected that pack with fewer arrows.
  + Actions for the attack are: **aaw** - up, **aas** - down **aaa** - left, **aad** - right, **Aawa** - mountains left **aawd** - up right, **aasa** - down left **aasd** - bottom right.

* **Defence mechanism - Shield**
  + If the opponent carries a shield with him at the time of the attack (with an arrow or a sword), they do not lower its HP. Each shield has its 30HP, that means when an opponent attacks a player with a shield, they will reduce HP to that shield. In case the shield has less HP than it needs to be reduced away, points will be reduced first to him and then to the player. If a player has two shield with himself, during the attack it will decrease HP to the shield that has less HP.
* **Shield release**
  + The player can remove the shield he owns at the moment with him.
  + Action drain shield: **dsh** .

## Health points of the buildings

Each building has a certain number of HP, and the demolition of the opponent's buildings brings the player points:

* House: 40 HP. Demolishing an opponent's house brings 40 points.
* Fortress : 50 HP. Demolishing an opponent's fortress brings 50 points.
* The fortress of shields: 60 HP. Demolition of the fortress for the enemy shields brings 60 points.
* The fortress of swords: 60 HP. Demolishing an opponent's sword fortress brings up 60 points.
* The fortress of the arrow : 60 HP. Demolition of the fortress for the arrows of the enemy brings 60 points.

## The avatars death

With the death of the Avatar, the position of both players is restarted to the starting position, and when the avatar is killed he will lose one life. Taking an opponents life gives you 300 points.

## Conditions in case of victory

If the Avatar of a particular team dies 5 times, the opposing team wins and the game ends.

If after a certain number of moves where there is not any avatar that has lost all 5 lives, the winner is the one who scored more points. If the score is equal, the winner is the one who has more HP.

## Score

For each game there the score of each player is saved.

# 

# How to program for AIBG

You can use any language that can handle API calls, but in conditions to start the server, you must have JDK (or JRE) installed for Java 8 or later.

## Install Java for ubuntu (Linux)

[http://bit.ly/aibg-install-java-linux](https://translate.google.com/translate?hl=sr&prev=_t&sl=sr&tl=en&u=http://bit.ly/aibg-install-java-linux)

$ sudo apt update

# check if Java is installed:

$ java -version

# if the output is “Command 'java' not found, but can be installed  
# with:…“ follow the instuctions

$ sudo apt install default-jre

# after doing the previous step check if installed coreclly:

$ java -version

## Java installation for Windows

[http://bit.ly/aibg-install-java-windows](https://translate.google.com/translate?hl=sr&prev=_t&sl=sr&tl=en&u=http://bit.ly/aibg-install-java-windows)

**Step 1: Download JDK**

1. Go to Java SE download webpage:  <http://www.oracle.com/technetwork/java/javase/downloads/index.html>.
2. Under  "Java Platform, Standard Edition" click on download button
3. Mark "Accept License Agreement" under the version you have downloaded
4. Choose the OS you are using
5. Install the downloaded file
6. Add the path to Java in PATH variable:  
   <https://docs.alfresco.com/4.2/tasks/fot-addpath.html>

Path:

JDK: "C:\Program Files\Java\jdk-{x}", where {x} is installed the version of the Java

JRE: "C:\Program Files\Java\jre-{x}“, where {x} is installed the version of the Java

# Instructions for starting the game during programming and testing

You will get the whole project to run on your laptop.

**Bot testing:**

1. Position yourself in the project directory
2. Position yourself in the build directory
3. Run the java -jar build.jar
4. You can play alone against your bot or make two different bots (you are doing everything on your own laptop)

**Endpoints:**

1. /train/random?playerId=XX - creates a random game and immediately inserts the first player and waits other player to make the same call to: /train/random?playerId=XX
2. /game/play?playerId=XX&gameId=XX connects to a specific gameId with specific playerId
3. /admin/createGame?gameId=XX&playerOne=XX&playerTwo=XX&mapName=mapConfigXX creates a game ready to connect via endpoint 2)
4. **/doAction?playerId=XX&gameId=XX&action=XX here bot sends commands to correct a particular game**
5. /?gameId=XX opens the browser to watch the game live

**Note:**

The move timeout is 300 seconds during programming. It will be 1 second during the competition.

# JSON

JSON example:

<https://docs.google.com/document/d/18Om3HG4BwdEX9waRlMo0K6T8J2rh7OecMPp9S33N1hM/edit?usp=sharing>

Within JSON, nextPlayer can be 0 or 1. Indicates whether player1 is next in play or player2. PlayerIndex indicates which player you are within JSON player1 or player2.

# STRUCTURE OF COMPETITION

Teams will be divided into four groups of three teams. Each team will fight with each other team in the group. Each one wins bring one point and a draw is not possible. First team from the group according to the number of wins or to *the goal differences* through to the *knockout* phase which will compete with the winners from other groups. In the *knockout* stage will be played the semi-finals, then a game for third place and at the end will be played the final (in the order).

# NOTE

Before the start of the programming necessary to leave your CV on [**cv.best.rs**](https://translate.google.com/translate?hl=sr&prev=_t&sl=sr&tl=en&u=http://cv.best.rs/).