Significant Algorithms

**Fighting** – when a fight request is accepted, fight method is being called taking opponent (Player object) as a parameter. Opponent’s and player’s current health values are being decreased in a loop until one of the values reaches 0. First, from opponent’s current health result of multiplying player’s current strength, complement of opponent’s current defence, and a random value between 0 and 1 is being subtracted, then from player’s current health result of multiplying opponent’s current strength, complement of player’s current defence, and a random value between 0 and 1 is being subtracted. Once one of the values reaches 0, opponent’s current health is being returned. If the returned value is larger than 0, the fight is won by the opponent, and player gets message about losing. Otherwise, the fight is lost by the opponent and player gets message about winning and gets money reward.

**Breeding** – number of children being result of breeding is calculated by multiplication of square root of product of both parents’ fertilities and constant indicating maximum number of children. Date of birth of each child is current date and date of death is current date increased by constant indicating lifespan of a monster. Owner’s ID of each child is assigned to player’s ID, and name is generated using the name generator. For defence, strength, health, and fertility of each child, there is 5% chance of mutation (so it gets set to a random value between 0 and 1), and equal chance of inheriting the value from one or the other parent.

**Monster’s name generation** – name for a monster is generated by randomly choosing a string from an array of 25 names, and randomly choosing a string from an array of other 19 names, giving 475 unique combinations.

**Sorting high scores** – this is achieved by applying bubble sort algorithm in sortByMoney() method. If a player with higher place on the list has less money than the next one, the places on the list are being swapped.