Performance of Different Class and Race combination in *D&D 5e*

Chaowen Yang

A thesis submitted for the degree of Master of Science in Game Design

Supervisor: Dr. Spyros Samothrakis
School of Computer Science and Electronic Engineering
University of Essex

August 2018

Abstract

This dissertation will focus on a table role play game, *Dungeons and Dragons 5th edition*, and using programmers to simulate gameplay to find if there are any unbalanced elements in the game. Ability check, weapons' damage rate, and saving throw will be tested at the version v1.0. 12 Races and 14 Classes compose 168 different combinations will be tested at the programmer version v2.2, which analyses from three respects, exploration, social interaction, and combat. All code and test result can be found at the GitHub (link: https://github.com/fvictorique/CE901-D-D-5e-Balance-Test). For that important information, which will affect analyses, will be contained in the dissertation inside the paragraphs or at the index. The result shows that class *Fighter* and *Rogue*, race *Half-Orc*, *Halfling: Stout*, and *Halfling: Lightfoot* shows better performance than others. However, class *Wizard* and *Sorcerer*, race *Half-Elf*, *Human*, *Dragonborn*, and *Gnome: Forest Gnome* are weak at this test. The reason is varied and may because of the lack of the test system or may because of the class or race's unique capability. For the future, the programmer could be improved, by adding more functions and data; then the improved one might do better in analyzing the result and make it more believable.

Key words: Dungeons and Dragons, Dice, Table Role Playing Games

Content

1. Introduction

This dissertation will focus on a tabletop role-playing game *Dungeons & Dragons fifth edition (D&D 5e)*. It has influenced some games' design later. For example, in *Dungeons & Dragons(D&D)* serials, every player character has six main abilities, which are Strength (Str), Dexterity (Dex), Constitution (Con), Intelligence (Int), Wisdom (Wis) and Charisma (Cha). These abilities decide the player character characters' other properties like Ability Modify and Skills. Higher Ability value may help the character done a quest easier than the other one who has lower Ability number. However, a player character will not be allowed to good at all six abilities for the game balance and better gameplay experience, so that players need to adjust value between six abilities on their character's. Therefore, different strategies of configuration will generate different outcomes for a single character. After *D&D*, some games made base on its framework and added extra elements, then create a new game. *Final Fantasy XIV* is an example, the character in the game has five attributes which are Strength, Dexterity, Vitality, Intelligence, and Mind. These attributes will benefit different race's damage dealing, for example, Dexterity is the primary damage dealing stat for Bard, Ninja, and Machinist [1]. Not only in Ability part, but also on the Race and Class, they are similar with *D&D* as well. Therefore, to understand the model of *D&D* could be helpful for analyses other similar games which developed base on it.

For those who have not get a chance to play *D&D* before, this dissertation will briefly introduce the game with its history. In case to avoid misunderstanding later, this dissertation will focus on *Dungeons & Dragons fifth edition (D&D 5e)*, which released at August 2014. All rules and data chose to be analyses will take from it.

Moreover, this dissertation will collect and summarize some comments on the website to define an expected outcome of the result, by comparing and contrasting them. After the test, if differences exist, this dissertation will analyze the reason and try to find the key which causes the result.

This dissertation will use *C#* as coding language, *Visual Studio 2017* as Integrated Development Environment (IDE), and *Microsoft Excel* as a tool to manage data collected. In the early research, this dissertation wrote the version 1.0 program without realizing the complicated structure of the game, so that the logic of programming layout is simple. Latterly, this dissertation updated the version 1.0 to 2.0 and 2.2, which improved the logic by adding more class of coding layout to enhance its operability. Therefore, the version 2.2 could do more analyses than version 1.0. Both of the version 1.0 and 2.2 will be described in detail in section 5 and section 6.

For the analyses, *Microsoft Excel* is a convenience for handling data because it is easier for creating graphs and rank a list by using its functions. Therefore, this dissertation will evaluate data that collected from programmer version 1.0 and version 2.2 in section 7.

These two versions come out with some result, but due to the lack of time and some missing part like spells and magic, so that these test results are alternative. Players could use the result and take them into the consideration during the gameplay as a guide. However, these two programmers need to be improved for the better result, and at the end of this dissertation, this dissertation will list some shortcoming that needs to be enhanced in the future improvements.

2. Dice and Tools

2.1. Dice

This dissertation will mention dice many times in the rest parts. For easy and shot way to describe dice, this dissertation will reference article *Troll*, *a Language for Specifying Dice-rolls* by Mogensen [2] and use a similar approach to record dice. For example, roll ***n*** number dices, which has ***r*** side and add ***i*** modify will be written in this way:

$$nDr + i$$

For example, "*2D8 +3*" means roll two eight-side dices and add three. Moreover, D20 means twenty-side dice. Similarly, ***Dr*** means r-side dice.

2.2. Tools

This dissertation uses *C#* as a coding language and *Visual Studio 2017* as IDE. The main reason for using *C#* is this is a common coding language for students who study course about games, and most of them will learn how to use Unity or similar engine. Therefore, for the further study, if possible, next years' master students could keep improve this programmer and simulate better results. Moreover, it is easy to transfer to Unity to make a visual simulator, by copying the most code with some edition.

Microsoft Excel allows to add different tab inside a file, and it is easy to learn the formula in it so that it is. Moreover, it could export many formats so that the table result could be shared to the different format and can be opened via different software.

3. More About D&D

3.1. OD&D

Original Dungeons and Dragons (*OD&D*) was published in 1974 by *TSR* [3]. It motivated players and publishers; 3,000 copies sold, and some similar games made at that time [4]. Not complicated like the fifth edition, the original version only has three Classes and four races. In the latter years, supplements consequent on its trade.

3.2. Branch of *D&D*

In 1774, *TSR* produced two product lines for *D&D*; one is *Dungeons & Dragons Basic Set* (*BD&D*), the other one is *Advanced Dungeons & Dragons* (*AD&D*).

3.2.1. *AD&D* and *AD&D 2e*

For version *Advanced Dungeons & Dragons* (*AD&D*), there are three books were published between 1977 to 1985, which include *Monster Manual*, *Players Handbook* and *Dungeon Masters Guide (Core rulebooks complete)*. *AD&D* added more Classes, monsters, and other updates.

The *Advanced Dungeons & Dragon second edition* (*AD&D 2e*) includes a *Player's*

Handbook, *Dungeon Master's Guide* and *Monstrous Compendium* since 1989 to 1995. It changed the Combat system and introduced Armor Class (AC), which is the prototype of *D&D 5e*'s combat mechanic. In this version, character creation is modified as well, more flexible for Class choosing, higher max level and spells book.

3.2.2. *BD&D*

Not famous like the *AD&D*, *Dungeons & Dragons Basic Set* has less influence for *D&D* further games. However, it gives rules and character creation at the beginning levels so that it helps players who new to tabletop role-playing games [5].

3.3. *D&D 3e* and *D&D 3.5e*

Due to the lack of management and the product line is too bloated, *D&D* series was falling [6]. At that time, *Wizards of the Coast LLC* (simply refer as *WotC* or *Wizards*) purchased *TSR* [7]. Then *Wizards* combined *BD&D* and *AD&D* as a product line and published the third version of *D&D*, *Dungeons & Dragons 3rd edition* (*D&D 3e*). By using D20 system and enhanced combat system, *D&D 3e* attracted more players back to *D&D* [6].

In 2003, *Dungeons & Dragons 3.5 edition* (*D&D 3.5e*) was produced. *D&D 3.5e* is the updated version of *D&D 3e*. This edition modified unbalance part in the previous version, and it also improved the combat system and edited parameters about races and Classes [6].

3.4. *D&D 4e*

In 2008, computer games and video games were going to become the trend of games. *Wizards* changed some rules and cut some core settings to fit the trend. However, those changes caused unsatisfied of some old players. Some players modified *D&D 3.5e* rules and developed *Pathfinder Role-playing Game* [6]. Taking all influence of *D&D 4e*, *Wizards* published a new version of D&D a few years later.

3.5. *D&D 5e*

Dungeons and Dragons fifth edition (*D&D 5e*), which released at August 2014. It includes the basic knowledge a player needs for playing the game, such as rules for creating a character, Exploration, and Combat. *Wizards* put elements back, which cut off in *D&D 4e*. There are three books included, *Player's Handbook*, *Dungeon Master's Guide* and *Monster Manual*.

In this dissertation, we are going to use the *Player's Handbook* [12] as the primary reference to design the programmers and taking parameters.

4. Study Area and Expected Result

In this dissertation, the goal is to write programmers to simulate game player of *D&D 5e*. There are three cores of adventure in *D&D*, **Exploration**, **Social Interaction** and **Combat**. Exploration and Social Interaction require player character to do 1D20 check; Combat needs to simulate battle with enemies. Before analyses *D&D 5e* this dissertation will collect some comments of *D&D* 5*e* and infer a possible outcome for the result, it could be the expected result and motivate this dissertation working on it.

4.1. Three Analyses Area

D&D list 9 main Races and some of them has few subraces, so that entire 14 optional Races; 12 main Classes with multiple detail choice total 40 optional Classes. Different combination of Race and Class may vary outcomes of the result. Those combination cause difference of a player character's six Ability and they lean to distinct performance. However, due to the massive number of Class, and its com37MAGICplexity, **this dissertation only takes 14 Races and 12 main Classes as the consideration during the test**, so total 168 combinations will be tested.

To evaluating the performance of the combinations in **Exploration** and **Social Interaction**, the expression of the player character's 18 Skills is needed to be tested

and analyzed. Those Skills influenced by player character's six Ability Modify, Race bonus, and Class bonus.

For the Combat part, it is more complicated than the other two respects. Due to the lack of time and complex of *D&D* rules, this dissertation only focusses on physic damage, without spells nor magic. For Combat Test process, the player character and the enemy both need to roll 1D20 to decide the order of attack sequence. Then, the attacker needs to do Attack Check, which means roll 1D20 and add its modify, if the result equal or larger than defender's Armor Class, this Attack Check is passed. Once Attack Check is successful, the attacker will roll dice base on its equipment weapon's damage dice and add modify. More details will mention in section 5.

4.2. Background Research

By searching on the internet, there are some comments for *D&D 5e*. Here is a comment which lists *D&D 5e*'s Class into six groups via a user Sant [8] at goddessfantasy.net.

Environmental occupation:	Paladin
Secondary environment:	Barbarian, Fighter
Top:	Bard, Wizard, Warlock
Superior:	Druid, Ranger
Backbone:	Sorcerer, Monk, Rogue
Vulnerable:	Cleric

Environmental occupation: core, which fixes current game version, the most powerful Class
Secondary environment: second core, most team will contain one of those
Top: excellent at one area, and good at all other Class
Superior: excellent in one area but has apparent weaknesses
Backbone: good in some areas but not good enough with above Classes
Vulnerable: no unique Skills, standard Class

It shows that *Paladin* is the most powerful Class in all Classes, *Barbarian* and *Fighter* are also good at gameplay. However, Sorcerer, Monk, Rogue, and Cleric are weaker than other Classes during the gameplay.

Another user Flibbertigibbet [9] at rpg.net gave below answer for game balance

1. Fighter seems can do higher damage than most Martial Classes.
2. Barbarian, Ranger, and Warlock looked a little weak.
3. Paladin feature seems is good at Saving Throws

This comment show Flibbertigibbet's experience via a lot of gameplay. He holds the similar opinion with [8] that the *Fighter* and the *Paladin* are strong at the game, but he feels that Barbarian is not dominant like [8] shows.

If looking through one more comment on reddit.com wrote by uncleinabucket [10], he tells that

Rogue: Explosive in the early round, but moderate in the later rounds. Highest mobility and evasion
Warrior: Good and accurate damage, high Armor Class and Health Point
Paladin: Heal or damage dealer, cannot do both at the same round
Ranger: Like a fighter, and it can deal extra damage, but do more spellcasting in the later game
Wizards: All schools are more balanced in this version, players can easily keep up
Sorcerer: Less chance to be selected
Warlock: Many approaches can be used for playing this Class
Bard: It is the most improved Class, and its spellcasting is themed
Others: Omitted, because have not got chance to play them

Therefore, by looking through comments [8], [9], and [10], *D&D 5e* seems balanced, for they hold a different view for it, and none of them is super strong in all comments. The reason could be the different gameplay strategy, but the fact is players may use various strategies and approaches during the gameplay so that they have the different feeling for *D&D 5e*. It is a good chance to analyze it by using programmers objectively.

4.3. Expected Result

Most comments are based on the Class not Races, but it is enough to predict a result of the test. According to the research,

Fighter, *Paladin*, and *Barbarian* may have better performance for the test;

Cleric and *Sorcerer* may weaker than other class.

However, spells and magic are not yet been added to the programmer. Therefore, for Classes like *Wizard* and *Cleric* may be affected and punier than other Class.

5. Methodology

In this part, this dissertation will focus on methodology and explain how the process and logic. There is two version of the code in the GitHub, *version 1.0* and *version 2.2*. The previous version is for the general test, which does not consider the different character class and race combination. It simulated

1) How likely the different Ability Modify success to pass a quest in different modify level
2) 37 weapons' damage rate
3) Saving throw success chance

Due to the lack of design in the version 1.0, this dissertation rearranged the code by adding more classes to make the system more flexible. It improved the programmer so that we could analyze 168 different combinations of player characters' class and race in following aspects

1) How good the combination at Exploration
2) How good the combination at Social Interaction
3) How good the combination at Combat
4) How good the combination at all of above

5.1. Version 1.0

5.1.1. Success to Pass a Quest with Different Modify level

In this version, this dissertation tests the performance of different levels of Ability Modify, which defined by the player character's Ability. There are 11 levels of Ability Modify from -5 to +5, which shows in the *Table I*. When a player character doing Ability check, the player character will roll 1D20 and add tested Ability Modify to get a number, then compare the result with the quest level to judge if the player character pass it or not.

For example, if a player character is trying to do a Strength check to pick up a heavy store, with 13 Strength Ability. The Dungeon Master (DM), who lead the game keep going on, tells the player character that the quest level is *Hard Success* Level with threshold 20, which can be found in *Table II*. Then the player character needs to roll 1D20 and add 1, which comes from its Ability Modify (see *Table I*), as a result. For

instance, the player character is lucky and rolled 1D20 get 19, which is a huge number. Next, he will add 1 to 19 and get 20. The result 20 is equal to the threshold of Hard Success Level so that it means that the player character is just passing the quest. Moreover, in the [12], it mentions that the player will always get lose if 1D20 result is 1; the player will always get success if 1D20 result is 20.

Ability	Ability Modify		Ability	Ability Modify		Ability	Ability Modify
0-1	-5		8-9	-1		16-17	+3
2-3	-4		10-11	0		18-19	+4
4-5	-3		12-13	+1		20	+5
6-7	-2		14-15	+2			

Table I: Ability and Ability Modify

Difficulty Level	Threshold
Very Easy Success	5
Easy Success	10
Medium Success	15
Hard Success	20
Very Hard Success	25
Nearly Impossible Success	30

Table II: difficulty level and threshold

Above paragraph shows an example of one kind of Ability check, and another Ability Check is the same process. To get a better analyses outcome, this dissertation will use *C#* to write a function in *Visual Studio* to simulate dice rolls. Then, repeat the function to simulate dice rolling with large times and add different Ability Modify to see how good it will pass the different level of quests, and it could be explained as:

$$S_{ab} = \frac{\sum_1^t +1 \; if[(rdm_{20} + \; ab_{mod}) > \; T_q]}{t}$$

Where $\boldsymbol{S_{ab}}$ is the pass percentage of Ability Check

$\boldsymbol{t}$ is the total test times

$\boldsymbol{rdm_{20}}$ is a random dice between 1 to 20

$\boldsymbol{ab_{mod}}$ is the Ability Modify of the Ability $\boldsymbol{ab}$

$\boldsymbol{T_q}$ is the threshold of the quest $\boldsymbol{q}$

5.1.2. 37 Weapons' Damage Rate

In the *D&D*, player character could choose a lot of weapons. To find which weapon is worth to purchase in the total 37 weapons, this dissertation will use the version 1.0 to simulate the result.

A weapon's damage rate is relative to the weapon's damage dice, the player character's Ability Modify, and the enemy's Armor Class. For a weapon's damage dice, it is fixed and wrote on the rules book; for the player character's Ability Modify is changeable and it also depends on the different class and race bonus; for the enemy's Armor Class is varying base on the different enemy. Therefore, the function could be written as

$$D_w = \frac{\sum_1^{t_s} d_n * rdm_{d_r} + d_i}{t_s}$$

Where

$\boldsymbol{D_w}$ is the damage rate of the weapon $\boldsymbol{w}$

$\boldsymbol{d_n}$ is the weapon damage dice's $\boldsymbol{n}$

$\boldsymbol{rdm_{d_r}}$ is a random number between 0 to weapon damage dice's $\boldsymbol{r}$

$\boldsymbol{d_i}$ is the weapon damage dice's $\boldsymbol{i}$

$\boldsymbol{t_s}$ is the Attack Check test successful time, it comes from

$$t_s = \sum_0^t +1 \; if[(rdm_{20} + \; p_{abm}) \geq e_{ac}]$$

$\boldsymbol{t}$ is the total test times

$\boldsymbol{rdm_{20}}$ is a random dice between 1 to 20

$\boldsymbol{p_{abm}}$ is the player character's Ability Modify

Therefore, by setting a test time and weapon's list, it is easy to be written into a loop and test them one by one.

5.1.3. Saving Throw Success Chance

Saving throw will be rolled when the player character's health point equal to or less than zero. At that time, the player character has two counters, ***success*** and ***fail***. Then

the player character needs to roll D20 until the ***success*** or ***fail*** reach three counts. The rule count is based on below, and it is also mentioned in [12].

success +1 when $10 \leq rdm_{20} < 20$

success +2 when $rdm_{20} = 20$

fail + 1 when $rdm_{20} < 10$

fail + 2 when $rdm_{20} = 1$

Therefore, keep rolling dice when ***success*** or ***fail*** not get three and end it when one of ***success*** or ***fail*** has received three counters. If repeat this process again and again until a large number, then the result could be likely of how a player character passes the saving throw.

5.2. Version 2.2

In the version 2.2, more classes of coding layout added. More logic form makes the data could be saved and taken easier than the version 1.0. The version 2.2 is improved by the version 2.0, and it has fixed some bugs. Hence, this dissertation could reach the goal to analyze *D&D 5e*.

5.2.1. About the Player Class

To create a player character, we need to know two necessary gears, Class and Race. Different Class and Race has the different effect on the player character's Ability, proficiency weapons and proficiency armors. For example, Class *Barbarian* starts with weapons *Great Axe*, *Hand Axe*, and *Javelin*, but Class *Bard* begins with weapons *Rapier*, *Longsword*, and *Dagger*. Therefore, once Race and Class are choosing, the player character's Ability, weapons and armors will be added base on them. Moreover, every player character will have some point to insert into his or her Ability, and this dissertation will choose the approach two mentioned at [12], which requires to put value 15,14,13,12,10, and 8 into the six Ability. In the dissertation, these number will be added to the player character's Ability automatically; more detail will be explained in section 6. Next, calculate the player character's Skills, which shows in *Table III.*

abm.STR	***abm.DEX***	***abm.INT***	***abm.WIS***	***abm.Cha***
Athletics	*Acrobatics*	*Arcana*	*AnimalHandling*	*Deception*
	SleightofHand	*History*	*Insight*	*Intimidation*
	Stealth	*Nature*	*Medicine*	*Performance*
		Religion	*Perception*	*Persuasion*
		Investigation	*Survival*	

Table III: player character Ability modify influence (CON influences nothing)

Therefore, there are 18 Skills need to be tested total. However, not every Skill will influence the performance both of the Exploration and social interaction, the influence point of each Skills will be described detailly in section six.

5.2.2. How Good the Combination at Exploration

To analyze how good a player character at the Exploration area, it is necessary to calculate the result of every Skill and multiply with their influence point. It is similar to Ability Check in the version 1.0 but uses Skills, not Ability Modify. When a player character needs to do a Skill check, the player character needs to roll 1D20 and add the Skills value as a bonus. If the result equal to or larger than the threshold, the player character will pass it; otherwise the player character will fail.

Before it, we need a list, *Skills lis*t, to contain all Skills.

The formula could be written as

$$R_e = \sum_{s_i}^{s_{count}} r_{s_i} \times I_{s_i}$$

Where R_e is the result of the Exploration

$\boldsymbol{s_i}$ is the $\boldsymbol{i^{th}}$ Skill on the *Skills list*

$\boldsymbol{s_{count}}$ is the amount of the Skills

$\boldsymbol{I_{s_i}}$ is the influence point of $\boldsymbol{s_i}$

$\boldsymbol{r_{s_i}}$ is the result of $\boldsymbol{s_i}$ test, and it contains three test base on the player status. The following paragraphs will explain it in more detail.

Every player character has a status for the game; it includes *normal*, *advantage* and *disadvantage* three status. In the *normal* status, the player character can only roll dice once and take the result; in the *advantage* status, the player character needs to roll dice twice and take the larger one as the result; in the *disadvantage* status, player character also needs to roll dice twice but take the small one as the result. Therefore, $\boldsymbol{r_{s_i}}$ can be written as

$$r_{s_i} = rn_{s_i} \times I_{normal} + ra_{s_i} \times I_{advantage} + rd_{s_i} \times I_{disadvantage}$$

Where

$\boldsymbol{rn_{s_i}}$ is the result of $\boldsymbol{s_i}$ test when player character under the *normal* status

$\boldsymbol{I_{normal}}$ is the influence point of *normal* status

$\boldsymbol{ra_{s_i}}$ is the result of $\boldsymbol{s_i}$ test when player character under the *advantage* status

$\boldsymbol{I_{advantage}}$ is the influence point of *advantage* status

$\boldsymbol{rd_{s_i}}$ is the result of $\boldsymbol{s_i}$ test when player character under the *disadvantage* status

$\boldsymbol{I_{disadvantage}}$ is the influence point of *disadvantage* status

Same as the *Table II*, there are five levels of Skills check, *Very Easy Success*, *Easy Success*, *Medium Success*, *Medium Success*, *Very hard Success*, and *Nearly Impossible Success*. The thresholds of them are 5, 10, 15, 20, 25, and 30. Therefore $\boldsymbol{rn_{s_i}}$ can be wrote as:

$$rn_{s_i} = r_{ve} \times I_{ve} + r_e \times I_e + r_m \times I_m + r_h \times I_h + r_{vh} \times I_{vh} + r_{ni} \times I_{ni}$$

Where $\boldsymbol{r_{ve}}$ is the *Very Easy Success* test result check for $\boldsymbol{s_i}$

$\boldsymbol{I_{ve}}$ is the influence point for *Very Easy Success*

$\boldsymbol{r_e}$ is the *Easy Success* test result check for $\boldsymbol{s_i}$

$\boldsymbol{I_e}$ is the influence point for *Easy Success*

$\boldsymbol{r_m}$ is the *Medium Success* test result check for $\boldsymbol{s_i}$

$\boldsymbol{I_m}$ is the influence point for *Medium Success*

$\boldsymbol{r_h}$ is the *Medium Success* test result check for $\boldsymbol{s_i}$

$\boldsymbol{I_h}$ is the influence point for *Medium Success*

$\boldsymbol{r_{vh}}$ is the *Very Medium Success* test result check for $\boldsymbol{s_i}$

$\boldsymbol{I_{vh}}$ is the influence point for *Very Medium Success*

$\boldsymbol{r_{ni}}$ is the *Nearly Impossible Success* test result check for $\boldsymbol{s_i}$

$\boldsymbol{I_{ni}}$ is the influence point for *Nearly Impossible Success*

Test process of $\boldsymbol{r_{ve}}$, $\boldsymbol{r_e}$, $\boldsymbol{r_m}$, $\boldsymbol{r_h}$, $\boldsymbol{r_{vh}}$, and $\boldsymbol{r_{ni}}$ are same but with the different threshold, which can be found in *Table II* above. For example

$$r_{ve} = \sum_{0}^{t} +1 \; if[(rdm_{20} + \; m_{s_i}) \geq 5(th)]$$

Where $\boldsymbol{t}$ is the test time of each Skills

$\boldsymbol{rdm_{20}}$: a random dice between 1 to 20

$\boldsymbol{m_{s_i}}$ is the $\boldsymbol{s_i}$'s modify number

$\boldsymbol{th}$ is the threshold of the *Very Easy Success* test, and here is 5 for $\boldsymbol{r_{ve}}$

Similarly,

$$\boldsymbol{r_e} = \sum_{1}^{t} +1 \; if[(rdm_{20} + \; m_{s_i}) \geq 10] \quad \boldsymbol{r_m} = \sum_{1}^{t} +1 \; if[(rdm_{20} + \; m_{s_i}) \geq 15]$$

$$\boldsymbol{r_h} = \sum_{1}^{t} +1 \; if[(rdm_{20} + \; m_{s_i}) \geq 20] \quad \boldsymbol{r_{vh}} = \sum_{1}^{t} +1 \; if[(rdm_{20} + \; m_{s_i}) \geq 25]$$

$$r_{ni} = \sum_{1}^{t} +1 \; if[(rdm_{20} + \; m_{s_i}) \geq 30]$$

$\boldsymbol{ra_{s_i}}$ and $\boldsymbol{rd_{s_i}}$ are same with $\boldsymbol{rn_{s_i}}$.

$$ra_{s_i} = \; r_{ve} \times I_{ve} + r_e \times I_e + r_m \times I_m + r_h \times I_h + r_{vh} \times I_{vh} + r_{ni} \times I_{ni}$$

$$rd_{s_i} = \; r_{ve} \times I_{ve} + r_e \times I_e + r_m \times I_m + r_h \times I_h + r_{vh} \times I_{vh} + r_{ni} \times I_{ni}$$

However, $\boldsymbol{r_{ve}}$, $\boldsymbol{r_e}$, $\boldsymbol{r_m}$, $\boldsymbol{r_h}$, $\boldsymbol{r_{vh}}$, and $\boldsymbol{r_{ni}}$ used slightly different method to calculate it because the *advantage* and *disadvantage* status. There is an example for $\boldsymbol{r_{ve}}$ under the *advantage* status.

$$r_{ve} = \sum_{0}^{t} +1 \; if[(rdm_{20}^{larger} + \; m_{s_i}) \geq th$$

Where $\boldsymbol{rdm_{20}^{larger}}$ means choose the larger one from two random number, which from 1 to 20. It could be written as below|:

$$r_1 = \; rdm_{20}$$

$$r_2 = \; rdm_{20}$$

$$rdm_{20}^{larger} = r_1 \; if(r_1 > \; r_2), else \; r_2$$

For the disadvantage status, $\boldsymbol{r_{ve}}$ is

$$r_{ve} = \sum_{0}^{t} +1 \; if[\left(rdm_{20}^{smaller} + \; m_{s_i}\right) \geq 5(th)]$$

Where $\boldsymbol{rdm_{20}^{smaller}}$ means choose the smaller one from two random number, which from 1 to 20. It could be written as below|:

$$r_1 = \; rdm_{20}$$

$$r_2 = \; rdm_{20}$$

$$rdm_{20}^{smaller} = r_1 \; if(r_1 < \; r_2), else \; r_2$$

Therefore, $\boldsymbol{r_{ve}}$, $\boldsymbol{r_e}$, $\boldsymbol{r_m}$, $\boldsymbol{r_h}$, $\boldsymbol{r_{vh}}$, and $\boldsymbol{r_{ni}}$ for $\boldsymbol{ra_{s_i}}$ are:

$$r_{ve} = \sum_{0}^{t} +1 \; if[\left(rdm_{20}^{larger} + \; m_{s_i}\right) \geq 5] \;\; r_e = \sum_{0}^{t} +1 \; if[\left(rdm_{20}^{larger} + \; m_{s_i}\right) \geq 10]$$

$$r_n = \sum_{0}^{t} +1 \; if[\left(rdm_{20}^{larger} + \; m_{s_i}\right) \geq 15] \;\; r_h = \sum_{0}^{t} +1 \; if[\left(rdm_{20}^{larger} + \; m_{s_i}\right) \geq 20]$$

$$r_{vh} = \sum_{0}^{t} +1 \; if[\left(rdm_{20}^{larger} + \; m_{s_i}\right) \geq 25] \;\; r_{ni} = \sum_{0}^{t} +1 \; if[\left(rdm_{20}^{larger} + \; m_{s_i}\right) \geq 30]$$

$\boldsymbol{r_{ve}}$, $\boldsymbol{r_e}$, $\boldsymbol{r_m}$, $\boldsymbol{r_h}$, $\boldsymbol{r_{vh}}$, and $\boldsymbol{r_{ni}}$ for $\boldsymbol{rd_{s_i}}$ are:

$$r_{ve} = \sum_{0}^{t} +1 \; if[\left(rdm_{20}^{smaller} + \; m_{s_i}\right) \geq 5] \;\; r_e = \sum_{0}^{t} +1 \; if[\left(rdm_{20}^{smaller} + \; m_{s_i}\right) \geq 10]$$

$$r_n = \sum_{0}^{t} +1 \; if[\left(rdm_{20}^{smaller} + \; m_{s_i}\right) \geq 15] \;\; r_h = \sum_{0}^{t} +1 \; if[\left(rdm_{20}^{smaller} + \; m_{s_i}\right) \geq 20]$$

$$r_{vh} = \sum_{0}^{t} +1 \; if[\left(rdm_{20}^{smaller} + \; m_{s_i}\right) \geq 25] \;\; r_{ni} = \sum_{0}^{t} +1 \; if[\left(rdm_{20}^{smaller} + \; m_{s_i}\right) \geq 30]$$

5.2.3. How Good the Combination at Social Interaction

Social interaction used the same method above but changed influence point I_{s_i} in R_{si}. The formula is the same with exploration test.

$$R_{si} = \sum_{s_i}^{s_{count}} r_{s_i} \times I_{s_i}$$

Similarly, we are going to use these functions below as well

$$r_{s_i} = rn_{s_i} \times I_{normal} + ra_{s_i} \times I_{advantage} + rd_{s_i} \times I_{disadvantage}$$

$$rn_{s_i} = r_{ve} \times I_{ve} + r_e \times I_e + r_m \times I_m + r_h \times I_h + r_{vh} \times I_{vh} + r_{ni} \times I_{ni}$$

$$r_{ve} = \sum_0^t +1\ if[(rdm_{20} + m_{s_i}) \geq th]\ \ r_e = \sum_1^t +1\ if[(rdm_{20} + m_{s_i}) \geq 10]$$

$$r_m = \sum_1^t +1\ if[(rdm_{20} + m_{s_i}) \geq 15]\ \ r_h = \sum_1^t +1\ if[(rdm_{20} + m_{s_i}) \geq 20]$$

$$r_{vh} = \sum_1^t +1\ if[(rdm_{20} + m_{s_i}) \geq 25]\ \ r_{ni} = \sum_1^t +1\ if[(rdm_{20} + m_{s_i}) \geq 30]$$

$$ra_{s_i} = r_{ve} \times I_{ve} + r_e \times I_e + r_m \times I_m + r_h \times I_h + r_{vh} \times I_{vh} + r_{ni} \times I_{ni}$$

$$r_{ve} = \sum_0^t +1\ if[\left(rdm_{20}^{larger} + m_{s_i}\right) \geq 5]\ \ r_e = \sum_0^t +1\ if[\left(rdm_{20}^{larger} + m_{s_i}\right) \geq 10]$$

$$r_n = \sum_0^t +1\ if[\left(rdm_{20}^{larger} + m_{s_i}\right) \geq 15]\ \ r_h = \sum_0^t +1\ if[\left(rdm_{20}^{larger} + m_{s_i}\right) \geq 20]$$

$$r_{vh} = \sum_0^t +1\ if[\left(rdm_{20}^{larger} + m_{s_i}\right) \geq 25]\ \ r_{ni} = \sum_0^t +1\ if[\left(rdm_{20}^{larger} + m_{s_i}\right) \geq 30]$$

$$rd_{s_i} = r_{ve} \times I_{ve} + r_e \times I_e + r_m \times I_m + r_h \times I_h + r_{vh} \times I_{vh} + r_{ni} \times I_{ni}$$

$$r_{ve} = \sum_0^t +1\ if[\left(rdm_{20}^{smaller} + m_{s_i}\right) \geq 5]\ \ r_e = \sum_0^t +1\ if[\left(rdm_{20}^{smaller} + m_{s_i}\right) \geq 10]$$

$$r_n = \sum_0^t +1\ if[\left(rdm_{20}^{smaller} + m_{s_i}\right) \geq 15]\ \ r_h = \sum_0^t +1\ if[\left(rdm_{20}^{smaller} + m_{s_i}\right) \geq 20]$$

$$r_{vh} = \sum_0^t +1\ if[\left(rdm_{20}^{smaller} + m_{s_i}\right) \geq 25]\ \ r_{ni} = \sum_0^t +1\ if[\left(rdm_{20}^{smaller} + m_{s_i}\right) \geq 30]$$

Therefore, by using 18 Skills test check, we could analyze the performance of a player character in Exploration and Social Interaction areas. These results may vary between the different Race and Class.

5.2.4. About the Enemy Class

Before start talking about the Combat Test, there are some classes need to be claimed before it, and one of them is the enemy class. This class is similar to the player class but simpler. For the enemy only need to consider its ability, hit dice, Armor Class and weapon (or attack type).

Therefore, in the class, if determine an enemy's Ability, the class can automatically do the calculations to get the Ability Modify. Hit dice can be written as *nDr+i* , and it tells how much health point the enemy has. Hence, even two enemies have the same type and same name, but they may have different Health Point. Armor Class and weapon is not complicated than player class, and every enemy has its fixed Armor Class and weapon. Then, we can start the battle.

5.2.5. How Good the Combination at Combat

Combat Test is different from the other two because there need to add more class like weapon, armor and enemy to help this dissertation to do it. Next, this dissertation will explain it and describe the test processes.

Before it, we need a list, *enemy list*, which contains all enemies.

Then test result could be written as

$$R_c = \sum_{e_i}^{e_{count}} \sum_{t_0}^{t_t} r_{e_i}$$

Where $\boldsymbol{R_c}$ is the test result of the Combat

$\boldsymbol{e_i}$ is the i^{th}'s enemy in the *enemy list*

$\boldsymbol{e_{count}}$ is the amount of the *enemy list*

$\boldsymbol{t_s}$ is the test time for every enemy

$\boldsymbol{r_{e_j}}$ is the combat result with the enemy $\boldsymbol{e_i}$

This formula asks the player character battle with every enemy many times, and its

shows overall performance of the player character in the combat respect.

r_{e_j} could be illuminated by using a figure rather than an equation. The *Figure I* show the logic of the Combat Test and its process. At first, the player and the enemy both need to roll 1D20 to decide who attack first. When they get the same dice result, reroll it. Then, the larger one will attack first, if the defender's health point equal to or less than zero, the attacker gets the score. In the test, if the player character wins the battle, the player character will receive one point, otherwise, receive nothing.

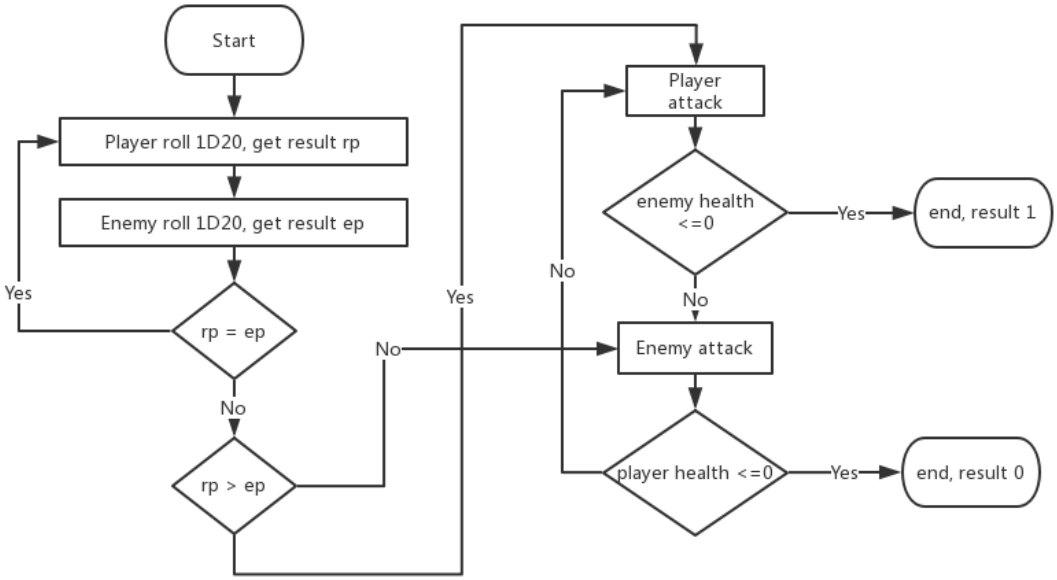


Figure I: Combat Process

Player attack and *Enemy attack both* need to use player class and enemy class. For the previous one, the player character needs to choose a weapon first, then do an *Attack Check*, which needs to roll 1D20 and plus the player character's modify to against the enemy's Armor Class. Once the *Attack Check* success, the player character could do *Damage Check*, which needs to roll the player character equipped weapon's damage dice and add the player character's modify. The same damage will be minus from the enemy's health point.

For the other one, the enemy needs to roll 1D20 plus its modify, *Attack Check*, to against the player character's Armor Class, which comes from the player character's equipped armor. If the enemy passes the *Attack Check*, then the enemy need to roll its damage dice plus its modify. Same as *the Player attack*, but the same damage will be deducted from the player character's health point.

By keep doing this loop, the final result could show the performance of the player character in the Combat area.

5.2.6. How Good the Combination for All Test

Once $\boldsymbol{R_e}$, $\boldsymbol{R_{si}}$, and $\boldsymbol{R_c}$ are calculated, add them together and times their individual influence point to get the final result. It could be explained in

$$R = R_e \times I_e + R_{si} \times I_{si} + R_c \times I_c$$

Where $\boldsymbol{R}$ means the final result of all test for

$\boldsymbol{I_e}$ is the influence point of the exploration test

$\boldsymbol{I_{si}}$ is the influence point of the social interaction test

$\boldsymbol{I_c}$ is the influence point of the Combat Test

The reason to use influence point is the $\boldsymbol{R_e}$, $\boldsymbol{R_{si}}$, and $\boldsymbol{R_c}$ come with the different number, and the differences may be very large. Hence, to adjust them to a similar level, which means the max value and the min value are closed.

Therefore, this dissertation could use $\boldsymbol{R}$, $\boldsymbol{R_e}$, $\boldsymbol{R_{si}}$, and $\boldsymbol{R_c}$ to do the analyses.

6. Implement code

Most code has explained inside the programmer, and some of them could understand its usage and function by reading the name. In this part, this dissertation will describe the approaches and codes used to achieve the goal. The code can be found *DnDProgrammerVersion2* at link:https://github.com/fvictorique/CE901-D-D-5e-Balance-Test

6.1. Version 1.0

6.1.1. Structure

In the version 1.0, there is no class for player and enemy yet. The variables could change six Abilities, Str, Dex, Con, Int, Wis, and Cha. There are two ways set abilities:

GetAbility()
or
SetAbility(int Str, int Dex, int Con, int Int, int Wis, int Cha)

The previous one will get the number the user inputted one by one in console interface; the latter one will set the number insider the code so that it is not flexible than the first way. Then, the user could add extra abilities bonus from class or race by using

ExtraAbility(int Str, int Dex, int Con, int Int, int Wis, int Cha)

Then, there is a method could update player character Ability Modify

CalculateMod();

An Ability will be minus 10 and get the floor result as this Ability Modify. For an ability smaller than 10, the ability will be minus 1 more to solve a bug. For example, if an ability is **15**, then the Ability Modify is $\frac{(15-10)}{2}$, and take the floor result of **2.5**, so the Ability Modify is **2**.

Then, ***CalculateSkills();*** is a function which will pass the different Ability Modify to the different Skills. If there are Skills bonus from the Race or Class, just use ***ExtraSkills(String skill, int proficiency)*** to add it. The user just needs to claim the name of the Skill and the bonus value, and the inside *switch* statement will match the Skill name and add value.

To set a player character's proficiency bonus, just pass the value to ***proficiencyBouns***.

proficiencyBouns = 2;

To add proficiency weapons or proficiency weapon types to the player character's proficiency weapons list, the user needs to add the weapon's name or types into a function, which with a *for* loop to add weapons. There are two examples of them:

AddProficiencyWeaponList(meleeWeaponList);
AddProficiencyWeaponList(new List<string>() {"weapon name"});

6.1.2 Success to Pass a Quest in Different Modify level

In this version, to find the influence for Ability Modify and the different quests pass rate under the normal status, the users will need following codes

```
SetAbility(0, 2, 4, 6, 8, 10); // modify -5, -4, -3, -2, -1, 0
CalculateMod();
Console.WriteLine("Normal status");
AbilityCheck(1000, NAD.Normal, NAD.Normal, NAD.Normal, NAD.Normal, NAD.Normal,
NAD.Normal);
SetAbility(12, 14, 16, 18, 20, 20); // modify 1, 2, 3, 4, 5, 5
CalculateMod();
AbilityCheck(1000, NAD.Normal, NAD.Normal, NAD.Normal, NAD.Normal, NAD.Normal,
NAD.Normal);
```

Where ***NAD*** is an enum of status which contains *NAD.Normal*, *NAD.Advantage*, *NAD.Disadvantage*.

AbilityCheck(int times, NAD Str, NAD Dex, NAD Con, NAD Int, NAD Wis, NAD Cha) is a function to do ability checker at ***times*** time(s), and abilities under the ***Str, Dex, Con, Int, Wis, Cha*** status in order.

Inside the ***AbilityCheck(int times, NAD Str, NAD Dex, NAD Con, NAD Int, NAD Wis, NAD Cha)***, it will run another function ***PrintAbilityCheck(String type, NAD nadState, int times)***, which will print the ***times*** time(s) Ability Checkresult of ***type*** under the ***nadState*** status. the latter function will show the ability's name, the Ability Modify the value, and the test result for *Very Easy*, *Easy, Medium, Hard, Very Hard,* and *Nearly Impossible* quests by using ***AbilityCheck(String t, int times, NAD nadstate, int difficultyLvl)***. The function will get the Ability's status and value by a *switch* statement, and use a *for* loop to test the successful times under the giving difficulty level, the result will be returned via some *if* statement.

Therefore, the logic can be illuminated in the Figure II at the appendix:

6.1.3. 37 Weapons' Damage Rate

AllWeaponCheck(String[] weaponList) is the function to calculate weapons damage rate, which formula shows in the section 5.1.2.. In this function, the user could choose what

kind of weapons to test. Version 1.0 has grouped the same type weapons into the different group for easier test: *meleeWeaponList*, *rangeWeaponList*, *martialMeleeWeaponList*, and *martialRangedWeaponList*. There are tree *for* loops inside the function: the first loop is for the list of *weaponList*; the second loop is for the player character's Ability, and it will start with 0 and end at 20 with 2 step every increment; the last loop is for the enemy's Armor Class. Hence, the function will test the relationship for a player character's Ability and enemies Armor Class with weapons in the list of *weaponList*.

The third loop used function ***GetWeaponDamageRate(int times, NAD status, string weapon, int ArmorClass)***, which used *if* statement to get the bonus, and pass values to get other variables such as the weapon's damage dice information. Then, work with ***TotalDamage(int times, NAD status, int diceNumber, int diceType, int modBouns, int weaponBouns, int armorClass)*** to get the total damage deal in ***times*** time(s). The latter function is formed with a *for* loop, an *if* statement, and another *for* loop. The deepest *for* loop is for the damage dice roll, in case ***n*** (in ***nDr + i***) larger than one; the *if* statement will run the function ***AttacCheck(NAD status, int bouns, int ArmorClass)*** to judge Attack Check pass or not; the cover *for* loop for time count.

AttacCheck(NAD status, int bouns, int ArmorClass) will return a boolean value base on the player character pass the Attack Check or not. It will random two value and determine which one will be used for returning by using *switch* and *if* statement.

Therefore, the logic of weapons' damage rate check could be illuminated in *Figure III* in the appendix.

6.1.4. Saving Throw Success Chance

The Saving Throw is used for check how likely a player character could revive when died, which rules is demonstrated in section 5.1.3.

The code in this part is simple. At the first, state two variables, *success* and *fail*. When none of them equal or larger than three, keep rolling 1D20 and add results into *success* or *fail*. If *success* equal to or larger than three, then this saving throw test is passed; otherwise this test is failed. Then, by using a *for* loop to this code, again and again, it will provide a percentage by using the total successful time divide by total test times. The logic could be illuminated in *Figure IV* in appendix.

6.2. Version 2.2

6.2.1. Improvement for Classes

In this version, there are some more classes added, *Dice*, *Status*, *Unit*, *Player*, *Ability*, *AbilityMod*, *Skills*, *EnemyAttack*, *Enemy*, *Weapon*, *Armor*, *ACMod*, *AttackChecker*, *UnityBase* and *EquipmentBase*.

Dice class is a class to handle dice and such information. The user could create a new type of dice by using ***new Dice("nDr + i")***. Inside the bracket, by change ***nDr + i*** to define a new dice type, and the class will handle information to separate int value, ***n***, ***r***, and ***i*** for later use.

Status class contains status information only, NAD means normal status, advantage status, and disadvantage status.

Unit class inherits *Status* class. It contains information about *health*, *maxhealth*, *Ability* class, *AbilityMod* class, and *armorClass*. This class defined the basic attributes for a unit and it could be inherited by *Player* class or *Enemy* class.

Player class inherits *Unity* class, and it is the most complex class in this version. It records information for a player character, *pRace* (player character's Race), *pCass* (player character's Class, to avoid key words class for *C#*), *primaryAbility*, *hitDice*, *Skills*, *proficiencyBonus*, an *EquipmentBase* class eb for equipment information, *weapon* and relative lists, *armor* and relative lists, and *ExtrallCapacity*. To create a

player character, just need to use ***Player(Race pRace, Class pClass, EquipmentBase eb, int proficiencyBouns)***, to set player character Race, player character Class, equipment (weapons and armors) information, and player character proficiency bonus. Then, the class will automatically set *Ability* for the player character's Ability base on the *primaryAbility*, calculate *AbilityMod,* calculate *Skills*, *hitDice*, *weapon*, and *armor*. Users could run ***GetCopy()*** insider the class to get a copy of this player character with the same *pRace*, *pClass*, *EquipmentBase*, and *proficiencyBouns,* however, it will automatically recalculate other information such as *Ability*, *AbilityMod*, *Skills*, a new random *weapon*, and a new random *armor*.

Ability class includes *STR*, *DEX*, *CON*, *INT*, *WIS*, and *CHA* for ability.

AbilityMod class includes Ability Modify for *STR*, *DEX*, *CON*, *INT*, *WIS*, and *CHA*. To set an *AbilityMod* class, an *Ability* class is needed, because the code needs to run base on it. Then, the code will do the same method as ***CalculateMod()***, which shows in section 6.1.1, and calculate values for Ability Modify.

Skills class is designed to calculate 18 Skills value automatically and a Skill Check function. To set *Skills*, the user needs to input *AbilityMod*.

EnemyAttack class is for *Enemy* use, and it concludes two type of attack mode, melee attack and ranged attack. To set it, just need to run ***EnemyAttack(Dice meleeAttack, int meleeAttackHitModify, Dice rangedAttack, int rangedAttackHitModify, int modify)***, where *Dice meleeAttack and Dice rangedAttack* used *Dice* class. If an enemy only has one attack mode, just need to set the other dice to "0D0 + 0".

Enemy class is similar to *Player* class but simpler. It inherits *Unit* class and contains variables for the enemy like *name* and *hitDice*.

Weapon class requires *name*, *type* (*SimpleMeleeWeapon, SimpleRangedWeapon,*

MartialMeleeWeapon, or *MartialRangedWeapon*), *damageDice*, *range* (*Melee* or*Ranged*). And *isFiness* information for a weapon.

Similar with *Weapon* class, *Armor* class require information like *name*, but some of them are different such as *type* (*LightArmor*, *MediumArmor*, *HeavyArmor* or *Shield*), *armorClass*, an *ACMod* (Armor Class modify information), *StrNeed* (strength need to wear it), a status for *Stealth*.

ACMod class is work with *Armor* class. It tells if the weapon's *armorClass* has Dexterity Modify bonus, and how much it is if it has.

AttackChecker class, contains two Boolean variables, one for recording if the Attack Check success or not, and the other one recording critical or not. This benefit for returning the result in a function together.

UnityBase class is a place contains player characters and enemies information. Inside the class, it contains two lists, one for players, and the other one for enemies.

EquipmentBase is a similar class with *UnityBase* class but contains information for weapons and armors. Two lists in the class, one for weapons, and the other one for armors.

6.2.2. Exploration Test

The test process is same which shows in the section 5.2.2. Before every single test, it is better to get a copy of the current test player character. Due to the improved classes, this version could do it by running code

```
Player copy = player.GetCopy();
exploratiionResult += copy.skills.SkillsCheck(ExplorationSkillsInfluence, 1, copy);
```

Inside the ***SkillsCheck(double[] checkList, int times, Player p)***, it puts the copy of the player

character's Skills into a list in order and creates another list for the different Skills influence point. Then, in the *for* loop, it is easy to manage by giving a number of indices.

SkillsCheck(double[] checkList, int times, Player p) use a *for* loop for counter test time, and add all result together to get the result for $\boldsymbol{R_e}$. Every Skill will be tested under the normal status, advantage status, and disadvantage status. These influence point $\boldsymbol{I_{advantage}}$, $\boldsymbol{I_{normal}}$, and $\boldsymbol{I_{disadvantage}}$ are 3, 8, and 3. It means most test is under the *normal* status, and some test will under the *advantage* or *disadvantage* status, which make sense for the gameplay. It uses another function ***SkillCheck(int skill, int times, Player p)*** to test the single Skill.

SkillCheck(int skill, int times, Player p) will test the pass rate for this Skill, and multiple the different influence point for the different levels quests. Then, once the *for* loop end, add all Skills' result together to get $\boldsymbol{R_e}$. The logic could be demonstrated in *Figure V* in the appendix.

6.2.3. Social Interaction Test

Social Interaction test is same with the Exploration test but with the different influence point. Therefore, the result is calculated in the same approach as section 6.2.2 but different influence point.

By using the same way,

```
Player copy = player.GetCopy();
socialInteractionResult += copy.skills.SkillsCheck(SocialInteractionSkillsInfluence, 1, copy);
```

it is easy to get the result of the player character in the social interaction test.

6.2.4. Combat Test

Combat part is different and complete than the other two tests. It requires enemy join into the battle.

```
BattleCheck(1000, player, ub.enemies)
```

There is a list for all enemies in the *UnitBase* class. To get the result, just use a copy of the player character to battle with every single enemy's copy inside the list one by one and add result together.

```
Player pCopy = p.GetCopy();
Enemy eCopy = e.GetCopy();
SimulateBattle(pCopy, eCopy);
```

Every battle test, the copy of the player character and the copy of the enemy are varied because the health point and others variable will be automatically recalculated. Therefore, the test could be set to a random environment but in the limitation. Then, both copies roll dice to decide which one will attack first. Until one of them died (health point equal to or less than zero), they will attack each other in order. In the further test, due to the bug, the code limited the rounds in 40 rounds. Then the result is

```
result = (e.health <= 0)? 1: 0;
```

Then, the logic of Combat Test could be described as the Figure VI in the appendix.

6.3.5. Test Result

When finish exploration test, social interaction test, and Combat Test, time them with different influence point to get the result of the player character.

```
double result = 0;
exploratiionResult = exploratiionResult / 300;
result += exploratiionResult;
socialInteractionResult = socialInteractionResult / 300;
result += socialInteractionResult;
double combatResult = BattleCheck(1000, player, ub.enemies);
result += combatResult;
```

Similarly, by using a *for* loop, it is easier to test all combination of player character's Class and Race.

7. Result and Analyses

All result could be found at file *DnDTestResult.xlsx* in GitHub with link

https://github.com/fvictorique/CE901-D-D-5e-Balance-Test/blob/master/DnDTestResult.xlsx.

In this section, this dissertation shows the test result and describe them.

7.1. Ability Check Result

This dissertation test 10,000 times (10k), 1,000 times (1k), and 100 times to get the result which shows in *Table IV*. More detail information about it could be found at Github *DnDTestResult.xlsx*, label *MissionLvlTest*.

Normal	LVL \ MOD	-5	-4	-3	-2	-1	0	1	2	3	4	5
10k	- VeryEasy:	0.5	0.5514	0.6003	0.6514	0.7013	0.7502	0.7996	0.8495	0.9013	0.9476	1
	- Easy:	0.25	0.349	0.3996	0.3996	0.45	0.501	0.5505	0.5992	0.6483	0.6999	0.7489
	- Medium:	0	0.0507	0.0991	0.1495	0.2	0.2497	0.3007	0.352	0.402	0.4505	0.4999
	- Hard:	0	0	0	0	0	0	0.0502	0.1026	0.1492	0.1995	0.2489
	- VeryHard:	0	0	0	0	0	0	0	0	0	0	0
	- NearlyImpossible:	0	0	0	0	0	0	0	0	0	0	0
1k	- VeryEasy:	0.497	0.544	0.61	0.649	0.706	0.753	0.797	0.851	0.895	0.95	1
	- Easy:	0.25	0.296	0.345	0.401	0.459	0.395	0.548	0.6	0.652	0.699	0.754
	- Medium:	0	0.051	0.101	0.145	0.199	0.247	0.289	0.354	0.394	0.449	0.493
	- Hard:	0	0	0	0	0	0	0.049	0.101	0.145	0.205	0.251
	- VeryHard:	0	0	0	0	0	0	0	0	0	0	0
	- NearlyImpossible:	0	0	0	0	0	0	0	0	0	0	0
100	- VeryEasy:	0.48	0.57	0.58	0.64	0.73	0.75	0.82	0.85	0.91	0.94	1
	- Easy:	0.25	0.27	0.37	0.41	0.45	0.53	0.58	0.62	0.66	0.71	0.76
	- Medium:	0	0.06	0.11	0.17	0.17	0.25	0.3	0.32	0.36	0.44	0.48
	- Hard:	0	0	0	0	0	0	0.06	0.09	0.14	0.2	0.26
	- VeryHard:	0	0	0	0	0	0	0	0	0	0	0
	- NearlyImpossible:	0	0	0	0	0	0	0	0	0	0	0

Table IV: (Normal status)10k, 1k, 100 test result

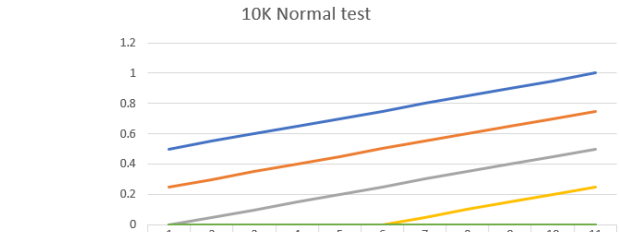
The result shows that 10k test is more approach to a stable rate and it should be enough for test. 1k and 100 test fewer times so that the deviation with 10k is large sometimes. Therefore, for the later test, this dissertation set test time as 10k. Then, using the same way, to get *Table V* and *Table VI* under the *advantage* status and *disadvantage* status.

Advantage	LVL \ MOD	-5	-4	-3	-2	-1	0	1	2	3	4	5
10k test	- VeryEasy:	0.7556	0.8064	0.8521	0.8958	0.9203	0.9388	0.965	0.9803	0.9882	0.9942	1
	- Easy:	0.4452	0.5982	0.6516	0.6516	0.6998	0.7542	0.8041	0.8501	0.8996	0.917	0.9433
	- Medium:	0	0.094	0.1883	0.2841	0.3629	0.4448	0.5238	0.5964	0.6471	0.6995	0.7561
	- Hard:	0	0	0	0	0	0	0.0935	0.1857	0.2826	0.3671	0.4485
	- VeryHard:	0	0	0	0	0	0	0	0	0	0	0
	- NearlyImpossible:	0	0	0	0	0	0	0	0	0	0	0

Table V: (Advantage status)10k, 1k, 100 test result

Disadvantage	LVL \ MOD	-5	-4	-3	-2	-1	0	1	2	3	4	5
10k test	- VeryEasy:	0.2483	0.2949	0.3451	0.3989	0.4797	0.5579	0.6357	0.7231	0.8067	0.9053	1
	- Easy:	0.0573	0.0788	0.0993	0.1437	0.1931	0.2449	0.2967	0.3598	0.4188	0.4919	0.562
	- Medium:	0	0.0057	0.0104	0.0153	0.0373	0.0549	0.0766	0.1244	0.1581	0.1997	0.2464
	- Hard:	0	0	0	0	0	0	0.0051	0.0105	0.0237	0.0413	0.064
	- VeryHard:	0	0	0	0	0	0	0	0	0	0	0
	- NearlyImpossible:	0	0	0	0	0	0	0	0	0	0	0

Table VI: (Disadvantage status)10k, 1k, 100 test result



	1	2	3	4	5	6	7	8	9	10	11
- VeryEasy:	0.5	0.5514	0.6003	0.6514	0.7013	0.7502	0.7996	0.8495	0.9013	0.9476	1
- Easy:	0.25	0.2966	0.349	0.3996	0.45	0.501	0.5505	0.5992	0.6483	0.6999	0.7489
- Medium:	0	0.0507	0.0991	0.1495	0.2	0.2497	0.3007	0.352	0.402	0.4505	0.4999
- Hard:	0	0	0	0	0	0	0.0502	0.1026	0.1492	0.1995	0.2489
- VeryHard:	0	0	0	0	0	0	0	0	0	0	0
- NearlyImpossible:	0	0	0	0	0	0	0	0	0	0	0

Figure VII: 10k Normal ability test

Taking data from the *Table IV*, we can draw a graph like the *Figure VII*, the relationship of Ability Modify and pass rate of the different difficult mission level is a linear relationship. Therefore, it could be written as $\boldsymbol{y = mx + c}$ and $\boldsymbol{m}$ could be calculated by using

$$m = \frac{\sum_i^N (x_i - \bar{x})(y_i - \bar{y})}{\sum_i^N (x_i - \bar{x})^2} \quad [8]$$

Where $\boldsymbol{m}$ is the slop

$\boldsymbol{N}$ is the number of data points

$\bar{\boldsymbol{x}}$ is the mean value of value of $\boldsymbol{x_i}$'s

$\bar{\boldsymbol{y}}$ is the mean value of value of $\boldsymbol{y_i}$'s

Use another function to calculate $\boldsymbol{c}$

$$c = \bar{y} - (m \times \bar{x}) \quad [8]$$

Then, for *veryEasy* quest under the *normal* status, the pass rate could be described as

$$y = 0.045x + 0.75$$

Where $\boldsymbol{x}$ is the player character's Ability Modify

$\boldsymbol{y}$ is the pass rate

By keep doing this process, we can calculate other difficulty level quest pass rate with different player character's Ability Modify under the *normal* status and get below linear relation functions.

veryEasy	*y*	=	*0.050*	*x*	+	*0.75*
easy	*y*	=	*0.050*	*x*	+	*0.50*
medium	*y*	=	*0.027*	*x*	+	*0.20*
hard	*y*	=	*0.025*	*x*	+	*0.07*
veryHard	*y*	=	*0*	*x*	+	*0.00*
nearlyImpossible	*y*	=	*0*	*x*	+	*0.00*

Then, using the same way to draw graph under the *advantage* and *disadvantage* status to get the *Figure VIII* and *Figure IX*. Details could be found at Label *MissionLvlTest* inside file *DnDTestResult.xlsx* in GitHub.

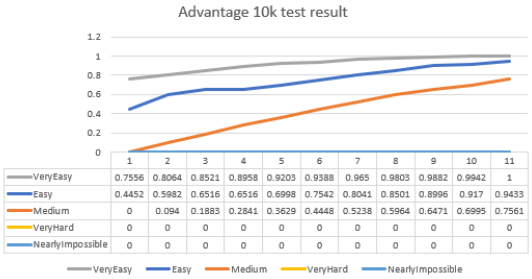


Figure VIII: 10k advantage ability test

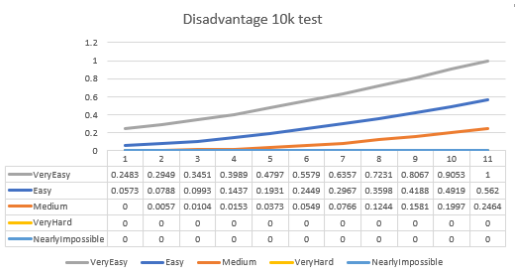


Figure IX: 10k disadvantage ability test

Then, calculate the linear relation functions for both statuses again, we can get *Table VII* below.

Normal	very easy	y	=	0.050	x	+	0.750
	easy	y	=	0.050	x	+	0.499
	Medium	y	=	0.027	x	+	0.205
	hard	y	=	0.025	x	+	0.068
	very hard	y	=	0.000	x	+	0.000
	Nearly impossible	y	=	0.000	x	+	0.000
average		y	=	0.025	x	+	0.254
Advantage	very easy	y	=	0.031	x	+	0.918
	easy	y	=	0.046	x	+	0.747
	Medium	y	=	0.076	x	+	0.418
	hard	y	=	0.046	x	+	0.125
	very hard	y	=	0.000	x	+	0.000
	Nearly impossible	y	=	0.000	x	+	0.000
average		y	=	0.033	x	+	0.368
Disadvantage	very easy	y	=	0.069	x	+	0.581
	easy	y	=	0.052	x	+	0.268
	Medium	y	=	0.025	x	+	0.084
	hard	y	=	0.005	x	+	0.013
	very hard	y	=	0.000	x	+	0.000
	Nearly impossible	y	=	0.000	x	+	0.000
average		y	=	0.025	x	+	0.158

Table VII: Linear Relationship in Different Status

By comparing and contrast data in *Table IV*, *Table V*, *Table VI* and *Table VII*, *advantage* status cause more pass rate than *normal* and *disadvantage* status. The reason is in the *advantage* status, the player character can roll dice twice and take the larger one as result, it makes sense for higher pass rate; in *disadvantage* status, the player character will take the lower result so that it will

For the *veryHard* difficult level quest and the *NearlyImpossible* difficult level quest, the player character cannot pass them at all status. It could be explained by the threshold are 25 and 30, therefore in this test, the largest value of 1D20 is 20 which is less than 25 and 30. Hence, without the bonus, a player has no chance to get a result which larger than 20. To fix it, in the next version, the designer adds a function to add the bonus, which comes from the player character's Ability Modify.

Generally, a player could use formulas in *Table VII* to calculate the pass rate for his player character during the gameplay. Just change $\boldsymbol{x}$ to a number which is the sum of the player character's Ability Modify and bonus, then get $\boldsymbol{y}$ as the percentage to manage strategy of gameplay.

7.2. Weapon damage result

In label of *meleeWeaponAfterDebug*, *RangedWeaponAfterDebug*, *MartialMelee*, and *MartialRange* in *DnDTestResult.xlsx* contain all type of weapons damage rate test result. Every single weapon has been tested in the different player character's Ability Modify and at the different enemy Armor Class. There is an example for a simple melee weapon *Club* (in label *meleeWeaponAfterDebug*) test result shows in *Table VIII*.

In *Table VIII*, the horizontal row, -5 to 5, is player character's Ability Modify or can be simplified describe as a bonus; the vertical column, 0 to 20, is enemy's Armor Class. To find the damage of weapon *Club* when a player character's bonus is 3 and the enemy's Armor Class is 6 and Health Point is 13, just need to find the column of 3 and row of 6 so that the damage rate is 5.3722. Then the player character might need to take

$\frac{13}{5.3722} \approx$ 2.4 rounds, so that take the ceiling value 3 and this player character may spend 3 rounds to defeat this enemy.

If the player character holds more than one weapons, by checking those table and compare the result could be an easy way to figure out which weapon might cause more damage under the circumstances.

	---------- Club ----------										
	-5	-4	-3	-2	-1	0	1	2	3	4	5
0	-2.87995	-1.7476	-0.63004	0.49687	1.6257	2.63081	3.62833	4.62543	5.62548	6.62453	7.62925
1	-3.00024	-1.87666	-0.7477	0.38193	1.50147	2.63099	3.62117	4.62764	5.62697	6.62325	7.61799
2	-3.12614	-1.99977	-0.87268	0.25251	1.37031	2.50657	3.62348	4.63023	5.61887	6.61492	7.61952
3	-3.25623	-2.1267	-1.00439	0.13054	1.25276	2.37032	3.49714	4.62861	5.62696	6.62531	7.62585
4	-3.36982	-2.2428	-1.13059	0.00518	1.12518	2.24921	3.37404	4.49693	5.62548	6.62805	7.62258
5	-3.49368	-2.37007	-1.25273	-0.11724	1.00104	2.12629	3.24494	4.37346	5.49883	6.62777	7.62567
6	-3.62426	-2.51057	-1.37989	-0.25132	0.87463	1.99687	3.12482	4.25252	5.3722	6.50795	7.62365
7	-3.74128	-2.6335	-1.49692	-0.37939	0.75997	1.87727	3.00634	4.12281	5.24827	6.37054	7.49813
8	-3.87622	-2.75215	-1.62521	-0.51147	0.61869	1.75287	2.88069	4.00241	5.12126	6.24913	7.37324
9	-4.00252	-2.88848	-1.74918	-0.62181	0.50393	1.62174	2.74367	3.87447	5.00582	6.12954	7.239
10	-4.12472	-3.00147	-1.87335	-0.74995	0.37049	1.49325	2.639	3.75274	4.87403	5.99325	7.12398
11	-4.24834	-3.12786	-2.00102	-0.87132	0.24139	1.36863	2.4955	3.62104	4.74583	5.86724	6.99492
12	-4.37992	-3.24724	-2.13091	-0.99936	0.12426	1.25046	2.37076	3.51052	4.62418	5.75061	6.87664
13	-4.49636	-3.37593	-2.25504	-1.12805	-0.00183	1.12801	2.24427	3.37164	4.49971	5.63205	6.74604
14	-4.61763	-3.50381	-2.38565	-1.24228	-0.12027	0.99248	2.12865	3.24436	4.386	5.50725	6.62107
15	-4.74982	-3.61904	-2.50972	-1.37147	-0.25002	0.88752	1.99521	3.13207	4.2546	5.38778	6.49787
16	-4.87319	-3.74927	-2.62285	-1.50465	-0.37538	0.74669	1.8865	3.00136	4.12989	5.25269	6.38488
17	-4.86993	-3.87563	-2.75029	-1.62776	-0.49916	0.62021	1.74779	2.87175	3.99409	5.12479	6.24909
18	-4.87609	-3.87311	-2.87719	-1.75162	-0.62675	0.49988	1.62708	2.75437	3.87318	5.00236	6.12156
19	-4.87266	-3.87365	-2.87429	-1.87656	-0.75037	0.36969	1.50014	2.62531	3.74094	4.88131	6.00388
20	-4.8743	-3.87804	-2.87516	-1.87142	-0.87268	0.2535	1.37423	2.49477	3.6265	4.75237	5.87876

Table VIII: Club Weapon Damage

Then, by using the same formula [8] shows in section 7.1

$$m = \frac{\sum_i^N (x_i - \bar{x})(y_i - \bar{y})}{\sum_i^N (x_i - \bar{x})^2}$$

and

$$c = \bar{y} - (m \times \bar{x})$$

to calculate the linear relationship for all weapons. However, due to the great data collected, find means of the weapons is necessary, before calculating linear relationship function. For instance, weapon *Club*, find means for damage as $\boldsymbol{x_i}$ and easy for later calculations.

xi	-5	-4	-3	-2	-1	0	1	2	3	4	5
yi	-4.1069682	-3.012425	-1.9111273	-0.8003927	0.31242545	1.42605727	2.53426136	3.63702	4.73268591	5.82512227	6.90788955

Table IX: Means of the weapon Club

By doing the same process, we can get the *Table X*. Larger ***m*** means how much damage in increasing number of player character bonus, larger ***c*** means the basic damage when without bonus

Name		Y	=	M	x	+	C
----------	Club ----------	y	=	1.26389884	x	+	1.41314079
----------	Dagger ----------	y	=	1.26400965	x	+	1.41333587
----------	Greatclub ----------	y	=	1.49711743	x	+	2.54458157
----------	Handaxe ----------	y	=	1.37768264	x	+	1.97826099
----------	Javelin ----------	y	=	1.3778673	x	+	1.97860008
----------	Light hammer ----------	y	=	1.26405669	x	+	1.41339508
----------	Mace ----------	y	=	1.37734603	x	+	1.97812599
----------	Quarterstaff ----------	y	=	1.37773459	x	+	1.9785907
----------	Sickle ----------	y	=	1.26375046	x	+	1.41332798
----------	Spear ----------	y	=	1.37796626	x	+	1.97889517

Table X: Simple Melee weapons' linear relationship

In *Table X*, weapon *Greatclub* has the largest ***m*** and the largest ***c***, which means this weapon could cause more damage than other simple melee weapons under the same circumstances. Therefore, based on the test, if a player character equips weapon *Greatclub*, it will bring more benefit than other simple melee weapons during the battle.

By using the same process, in *DnDTestResult.xlsx*, labels of *RangedWeaponAfterDebug*, *MartialMelee*, and *MartialRange* can leading-out *Table XI*, *Table XII*, and *Table XIII*.

Name		y	=	M	x	+	C
----------	Crossbow,light ----------	y	=	1.49632023	x	+	2.5430019
----------	Dart ----------	y	=	1.2635557	x	+	1.41297698
----------	Shortbow ----------	y	=	1.37811926	x	+	1.9783581
----------	Sling ----------	y	=	1.26370612	x	+	1.41350893

Table XI: Simple Ranged weapons' linear relationship

Table XI shows weapon *Crossbow,light* could deal more damage than other simple ranged weapons under the same situation.

Name		y	=	M	x	+	C
----------	Battleaxe ----------	y	=	1.49686	x	+	2.54427
----------	Flail ----------	y	=	1.49687	x	+	2.54403
----------	Glavie ----------	y	=	1.49666	x	+	2.54436
----------	GreatAxe ----------	y	=	1.74769	x	+	3.67265
----------	Greatsword ----------	y	=	1.80474	x	+	3.78982
----------	Battleaxe ----------	y	=	1.4969	x	+	2.54309
----------	Lance ----------	y	=	1.74813	x	+	3.67236
----------	Longsword ----------	y	=	1.4963	x	+	2.54387
----------	Maul ----------	y	=	1.8044	x	+	3.78974
----------	Morningstar ----------	y	=	1.49651	x	+	2.54418
----------	Pike ----------	y	=	1.49664	x	+	2.5439
----------	Rapier ----------	y	=	1.49641	x	+	2.54407
----------	Scimitar ----------	y	=	1.49694	x	+	2.54315
----------	Shortword ----------	y	=	1.37781	x	+	1.97853
----------	Trident ----------	y	=	1.37776	x	+	1.97876
----------	War pick ----------	y	=	1.49626	x	+	2.54382
----------	Warhammer ----------	y	=	1.49684	x	+	2.5441
----------	Whip ----------	y	=	1.26365	x	+	1.41319

Table XII: Martial Melee Weapons' Linear Relationship

Table XII tells that in the same conditions, weapon *Greatsword* could deals more damage than other martial melee weapons, but Whip will deals less damage than other martial melee weapons.

Name		y	=	M	x	+	C
----------	Blowgun ----------	y	=	1.04339	x	+	0.95455
----------	Crossbow,hand ----------	y	=	1.37739	x	+	1.9789
----------	Crossbow,heavy ----------	y	=	1.49685	x	+	2.54264
----------	Longbow ----------	y	=	1.49615	x	+	2.54378
----------	Net ----------	y	=	1	x	+	0

Table XIII: Martial Ranged Weapons' Linear Relationship

Table XIII shows weapon *Crossbow,heavy* could deal more damage when the player character's bonus increase; weapon *Longbow* could deal more damage when the player character has no bonus.

7.3. Saving Throw Test Result

By using the method described in section 5.1.3, and code in 6.1.4, this dissertation could get the result of saving throw below

Total test Times: *10000*
Total Success Times: *5960*
Total fail Times: *4040*

By using ***Total Success Times*** divide by ***Total test Times***, Saving Throw pass rate could be calculated. Therefore

$$\text{Saving Throw pass rate} = \frac{total\ success\ times}{total\ test\ times} = \frac{5960}{10000} = 59.6\%$$

It shows a player character's Saving Throw pass rate is not 50%, but higher 59.6%.

7.4. Race and Class Combination Test

This Race and Class combination test is based on the programmer version 2.2, which described at section 5.2 and section 6.2. The result of this test records at the label *all combination v2* in file *DnDTestResult.xlsx* in GitHub. The test result shows in *Table XIV* in the appendix.

7.4.1. Exploration and Social Interaction Test Result

Then, separate the *Table XIV* to four different tables, *Table XV* based on the result of the Total Test result, *Table XVI* bases on Exploration Test result, *Table XVII* bases on Social Interaction Test, and *Table XVIII* bases on Combat Test.

By skimming through the *Table XVI* and the *Table XVII*, for Exploration and Social Interaction test parts, *Halfling: Stout* and *Halfling: Lightfoot* are good at these two parts. By looking through label *AllTable* in *DnDTestResult.xlsx*, which contain all combination information, the same capacity can be found are

Dexterity +2
Reroll D20 when attack roll, Ability Check, or Saving Throw, once

The second capacity may be the reason why their scores are larger than other combinations, for they both get another chance to reroll dice one more time when they fail at the first time.

For the 10 lowest score in Exploration Test, Race *Human* occupy 10; for the 10 lowers score in Social Interaction Test, *Human* occupy 9. Therefore, *Human* is weak at both parts. By looking through *Human*'s capacity

Strength +1
Dexterity +1
Constitution +1
Intelligence +1
Wisdom +1
Charisma +1

These bonuses are very weak because it might not affect a player character's Ability Modify. For example, if a player character's Ability is 16, 14, 14, 12, 10, and 10 for *Str*, *Dex*, *Con*, *Int*, *Wis*, and *Cha*. Adding 1 more value for every ability, the player character's Ability will become 17, 15, 15, 13, 11, and 11. Visually, every Ability gets increasement, but the character's Ability Modify is same, for more detail about Ability Modify could be found at previous *Table I*. Hence, it could be a reason why *Human* did not good at these two parts.

Skimming through *Table XVI* and *Table XVII* again, the rule could be found that

***Race* influence more than *Class* in Exploration and Social Interaction Test**.

The reason could be this test evaluates a player character's Ability, higher Ability value will bring more benefits to the player character.

In these two parts, Race *Halfling: Stout*, Halfling: *Lightfoot*, *Dwarf: Mountain Dwarf*, *Half-Orc*, and *Dragonborn* have more benefits compare with other Races; Race *Human*, *Tiefling*, *Elf: Dark Elf*, *Half-Elf*, and *Tiefling*, have less bonus compare with other Races.

7.4.2. Combat Test Result

Combat Test rank result can be found at *Table XVII*. It shows the combination of *Fighter + Half-Orc* is much higher than other Class and Race combination, and higher 8.75% than the second highest combination. Class *Fighter* benefit a player character can be proficiency for all weapons and all armors, the hit dice of Fighter is D10 which is the second highest hit dice in all Class, just lower than D12 of Barbarian. Moreover, Race *Half-Orc* has capabilities

Relentless Endurance: When hit point is reduced to 0, back to 1 (CD: a long break)

Savage Attacks: add extra damage, if critical hit with a melee weapon

both of them benefit for combat. Therefore, if a player character's Race is *Half-Orc* or Class is *Fighter*, the player character's combat score will be improved.

Class *Roger*, *Paladin* and *Ranger*, Race *Halfling: Lightfoot* and *Halfling: Stout* also did a good job in the Combat Test. For the Class *Roger*, player character is proficiency for all equipped weapons at the beginning, and it could be a reason why its score is higher; for the class of *Paladin* and *Ranger*, both of them has higher hit dice, D10, than most other classes, it could benefit for combat because it causes a player character could take more damage before died. Race *Halfling: Lightfoot* and *Halfling* have the same capability

Reroll D20 when attack roll, Ability Check, or Saving Throw, once

It lets a player character reroll D20 when the first Attack Check fail, it makes the player character could have more chance to pass the Attack Check and deal damage to enemies.

However, for Race *Human*, *Half-Elf*, *Dragonborn*, and *Dwarf: Mountain Dwarf*, Class *Wizard*, *Sorcerer*, and *Monk* did not well in Combat Test. For Class *Human*, *Half-Elf*, *Dragonborn*, and *Dwarf: Mountain Dwarf*, they only improved some Abilities for a player character, not like Class *Halfling: Lightfoot* or *Halfling* have capabilities. For *Class* of *Wizard* and *Sorcerer*, they have the smallest hit dice, D6, which means they could take less damage before they died. For Class *Monk* has D8 hit dice but has no proficiency armors.

However, except for the top 20 and bottom 20, the middle 128 combinations have fewer differences and fewer variances. **Generally, Class *Roger*, *Paladin* and *Ranger*, Race *Halfling: Lightfoot* and *Halfling: Stout* are good at combat, but Race *Human*, *Half-Elf*, *Dragonborn*, and *Dwarf: Mountain Dwarf*, Class of *Wizard*, *Sorcerer*, and *Monk* are weak at this part.**

7.4.3. Total Test Result

Add result get from section 7.4.1 and section 7.4.2 together; this dissertation orders them together to get *Table XVIII*. **It shows that Race *Half-Orc*, *Halfling: Stout*, and *Halfling: Lightfoot* gets the better result than other Races, and most player character with Class *Fighter* or *Rogue* could get higher scores than other classes.** This result is similar with the result from section 7.4.1 and section 7.4.2 so that during the gameplay, **creating a character by choosing Class from *Fighter*, or *Rogue, and choosing Race from Half-Orc*, *Halfling: Stout*, and *Halfling: Lightfoot* could get better performance than other player characters.**

8. Discussion and Conclusion

This dissertation has written two programs to test the balance in *D&D 5e*. Two version of programmers were designed to test *D&D 5e* at different aspects. Via the version 1.0 and version 2.2, we could get the result that

1) tables for Ability Check pass rate under the *normal* status
   Figure VII, *Table IV* and *Table VII*
2) tables for Ability Check pass rate under the *advantage* status
   Figure VIII and *Table V*
3) tables for Ability Check pass rate under the *disadvantage* status
   Figure IX and *Table VI*
4) simple melee weapons' damage rate
   Table X
5) simple ranged weapons' damage rate
   Table XI
6) martial melee weapons' damage rate
   Table XII
7) martial ranged weapons' damage rate
   Table XIII
8) saving throw pass rate
   59.6% (10,000 test)
10) 168 race and class combination test result
   Table XIV, *Table XV*, *Table XVI*, *Table XVII* and *Table XVIII*

The result of 10) shows in *Table XIX* below

	Class	**Race**
Good performance	*Fighter* and *Rogue*	*Half-Orc*, *Halfling: Stout*, and *Halfling: Lightfoot*
Bad performance	*Wizard* and *Sorcerer*	*Half-Elf*, *Human*, *Dragonborn* and *Gnome: Forest Gnome*

Table XIX: Unbalance Classes and Races During the Test

The test illustrates that not every class and race is fair and balanced; it could influence gameplay for players. However, it also may because of the lack of these two programs. Therefore, in the future, there are some places to be improved to make these programs better and result more reliable.

1) Add spells and magic to the player character

2) Add all enemies, because for current version, only half enemies added for testing

3) Add a level system for the player character

4) Improve influence point for exploration test and social interaction test

5) Improve combat progress

6) Make a character choose weapons and armors base on the test result, not random

7) Modify influence point when need

8) Simplify code and fix bugs when encountered

9. References

[1] "Attributes - Final Fantasy XIV A Realm Reborn Wiki - FFXIV / FF14 ARR Community Wiki and Guide", *Ffxiv.consolegameswiki.com*, 2018. [Online]. Available: https://ffxiv.consolegameswiki.com/wiki/Attributes. [Accessed: 05- Aug- 2018].

[2] T. Mogensen, *Troll, a Language for Specifying Dice-rolls*. ACM Symposium on Applied Computing, 2009, pp.1910-1915

[3] " Birth of Dungeons & Dragons and its prime development (2)", *G-cores.com*, 2017. [Online]. Available: https://www.g-cores.com/articles/93885. [Accessed: 07- Aug- 2018].

[4] J. Peterson, *Playing at the world*. 2012, p. 496.

[5] D.Cowie, "Game Reviews", *Imagine*, p42. ,1983

[6] wangxiexie, "Dungeons & Dragons shattered crisis: The thriving TSR company", *G-cores.com*, 2017. [Online]. Available: https://www.g-cores.com/articles/94156. [Accessed: 07- Aug- 2018].

[7] S. Appelcline, "A BRIEF HISTORY OF GAME #1: WIZARDS OF THE COAST: 1990-PRESENT", *rpg.net*, 2006. [Online]. Available: https://www.rpg.net/columns/briefhistory/briefhistory1.phtml . [Accessed: 07- Aug- 2018].

[8] Sant, "Five editions of the latest combat rating", *Goddessfantasy.net*, 2017. [Online]. Available: http://www.goddessfantasy.net/bbs/index.php?topic=96989.0. [Accessed: 21- Aug- 2018].

[9] Flibbertigibbet, "Does the Class Balance of the 5E PHB Core Classes stay close all the way from Level 1 to Level 20? - Page 2", *RPGnet Forums*, 2017. [Online]. Available: https://forum.rpg.net/showthread.php?801574-Does-the-Class-Balance-of-the-5E-PHB-Core-Classes-stay-close-all-the-way-from-Level-1-to-Level-20/page2. [Accessed: 20- Aug- 2018].

[10] uncleinabucket, "[5E] Class Balance - make me feel better", *reddit*, 2015. [Online]. Available: https://www.reddit.com/r/DnD/comments/2erd88/5e_class_balance_make_20me_feel_better/. [Accessed: 20- Aug- 2018].

[11] M.Gardner, "CE902 Spring Term Lecture 4: Correlation and Linear Regression ", University of Essex, 2018

[12] M. Mearls, J. Crawford, *D&D Player's Handbook,5th edition*, U.S:Hasbro SA, 2014

10. Appendix

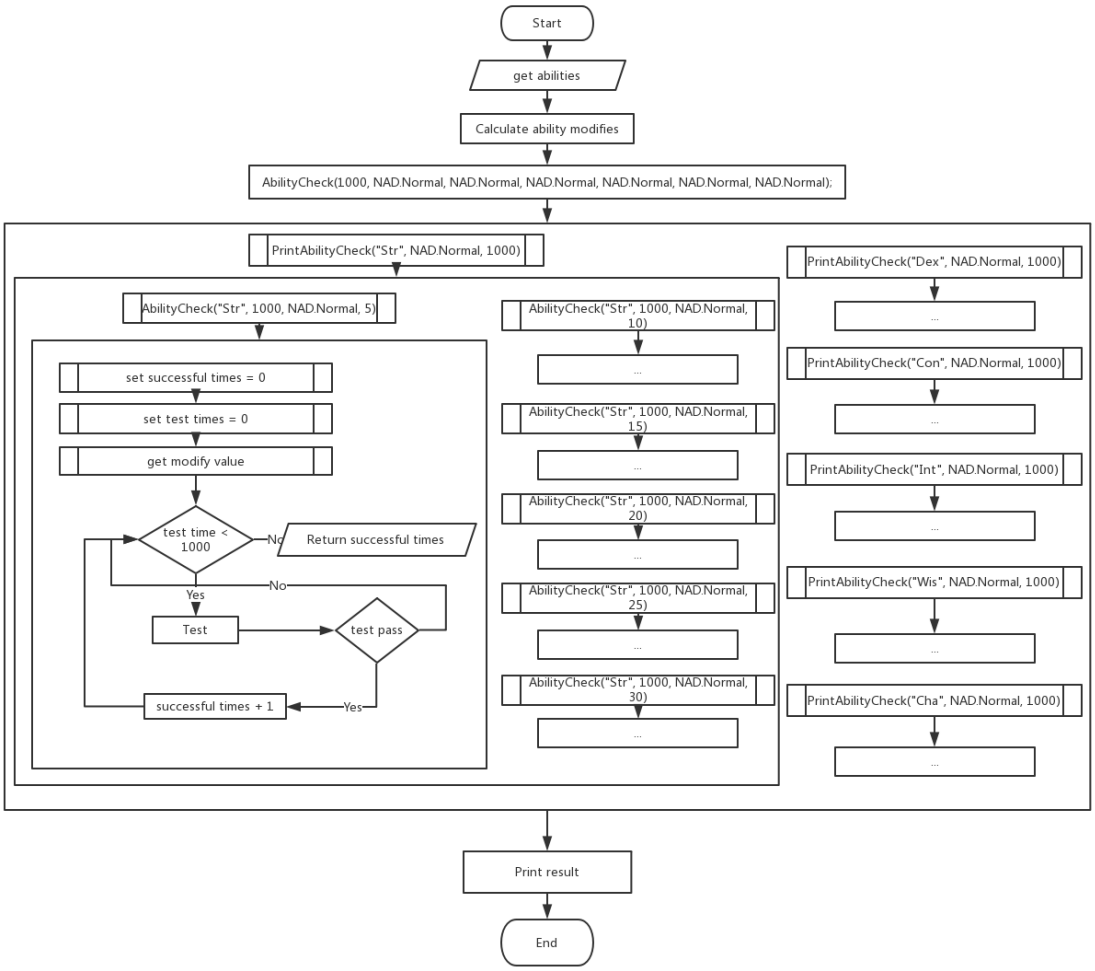


Figure II: Ability Check Code Logic

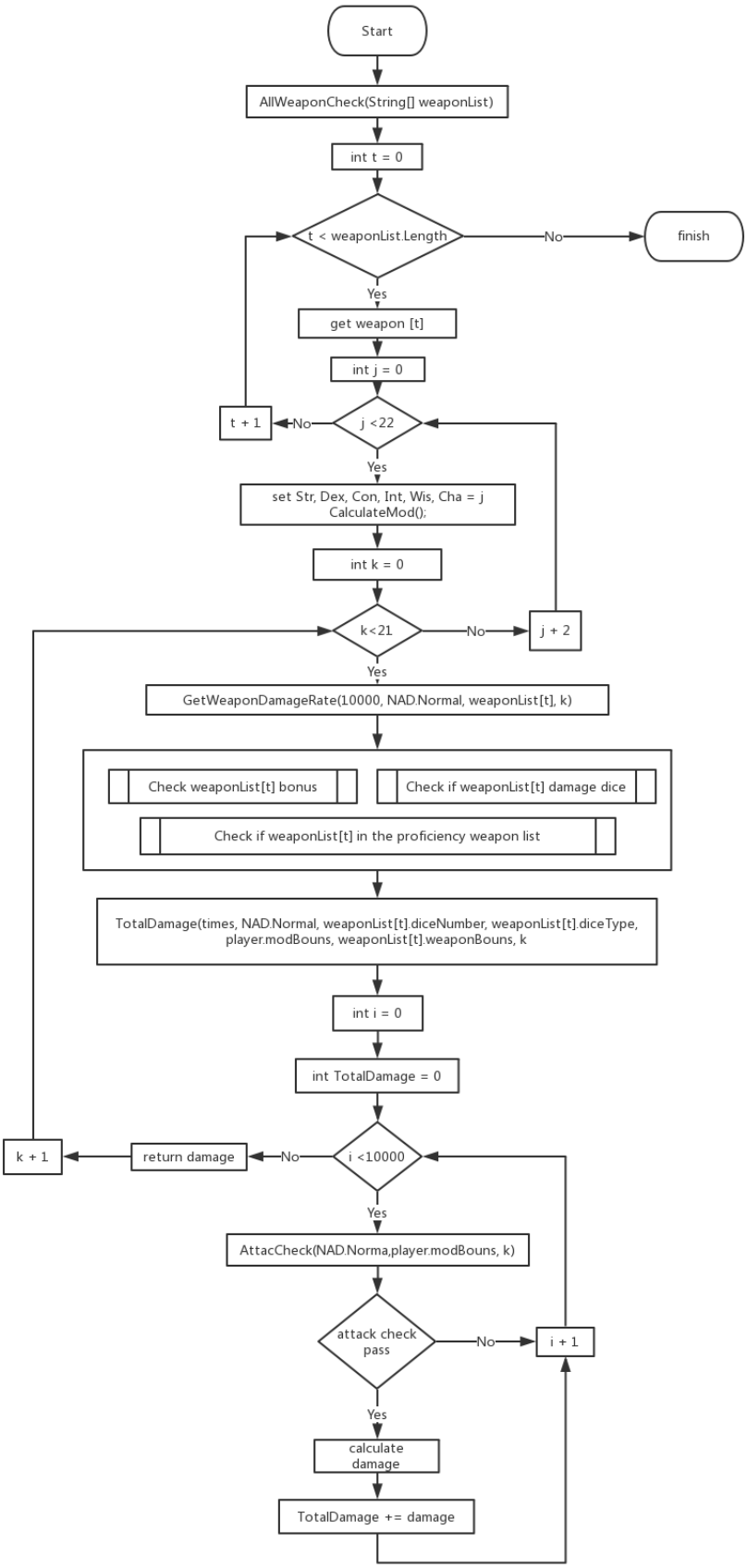


Figure III: Weapons' Damage Rate Check Process

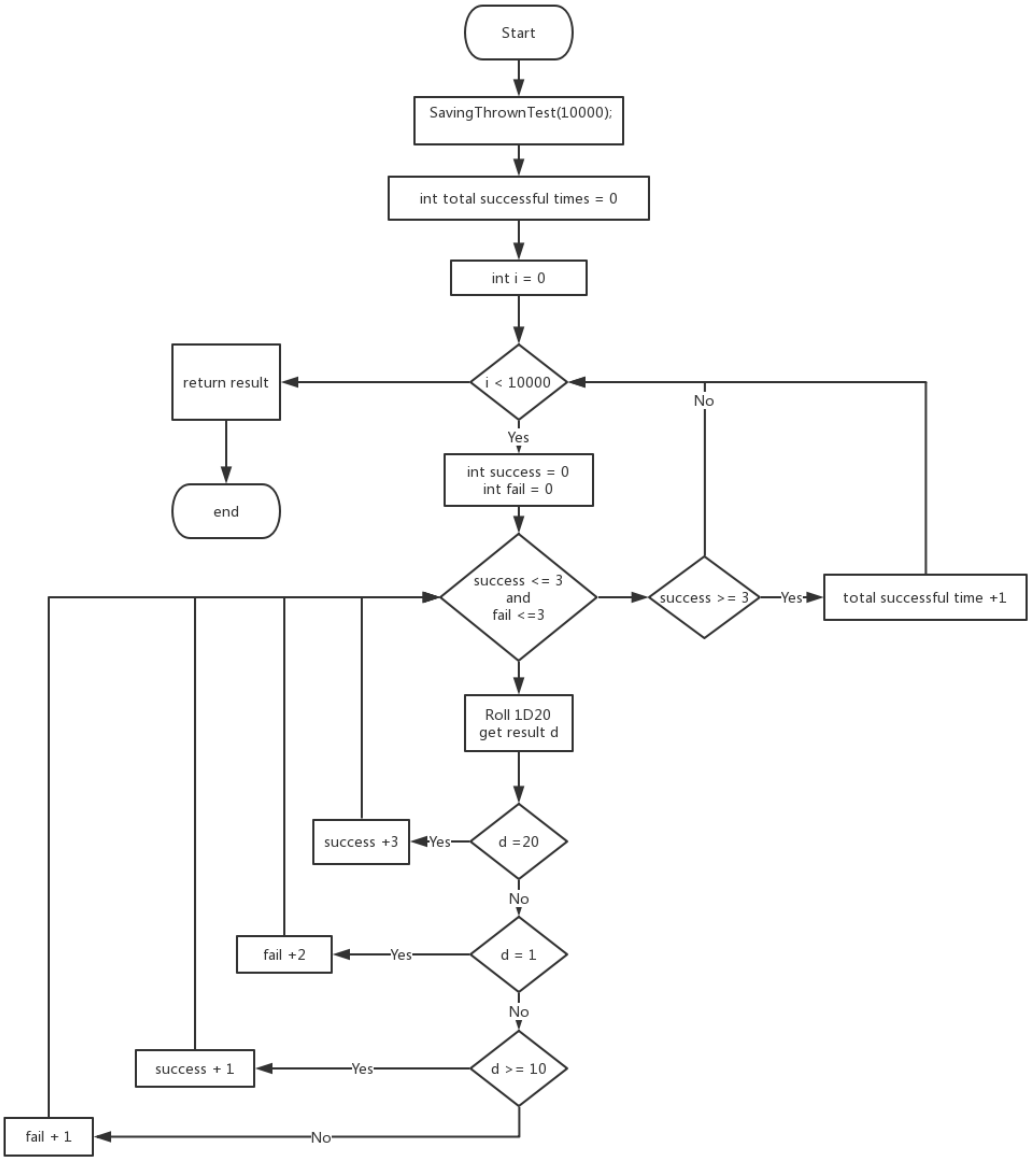


Figure IV: Saving Throw Test Process

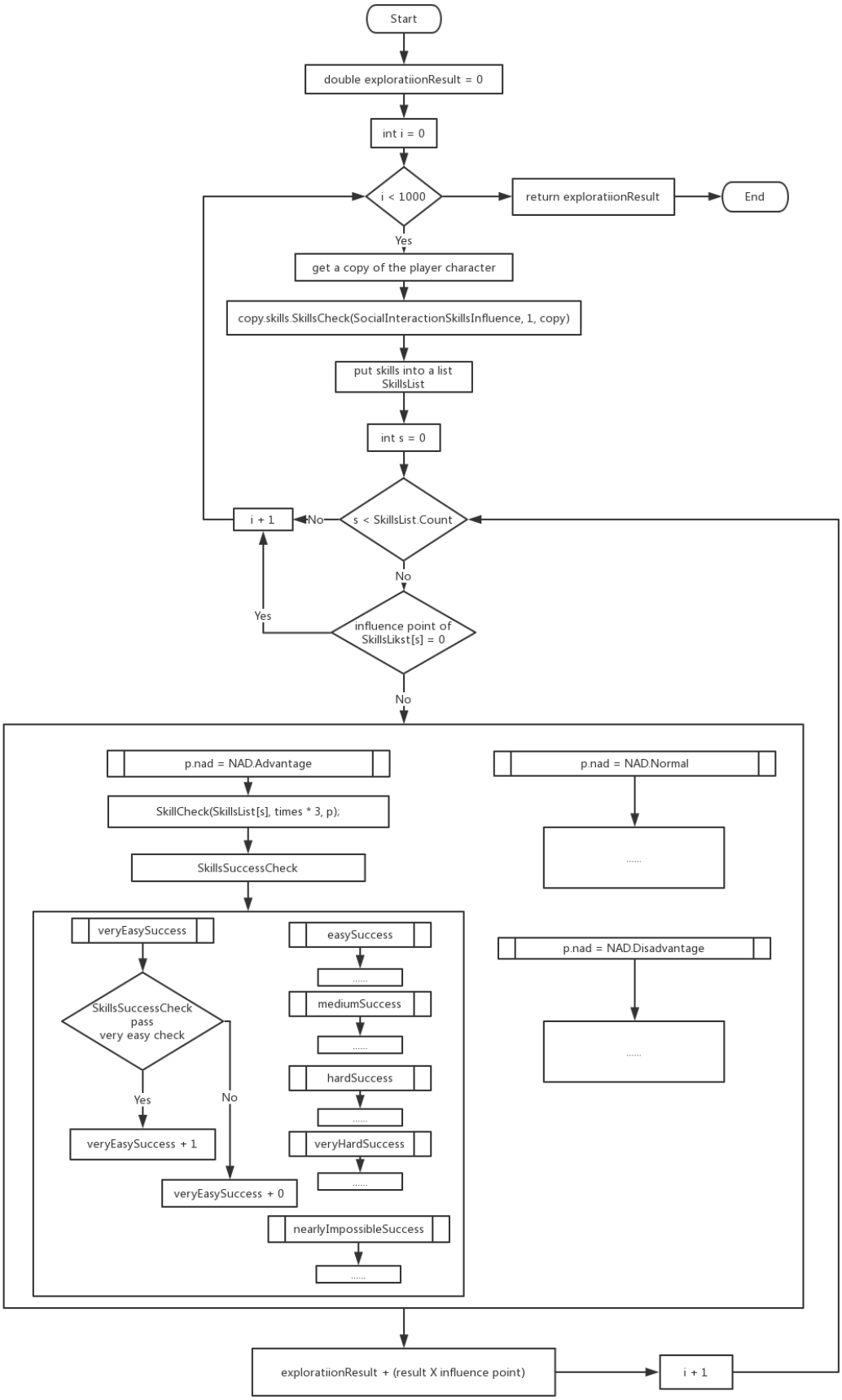


Figure V: Exploration Test Process

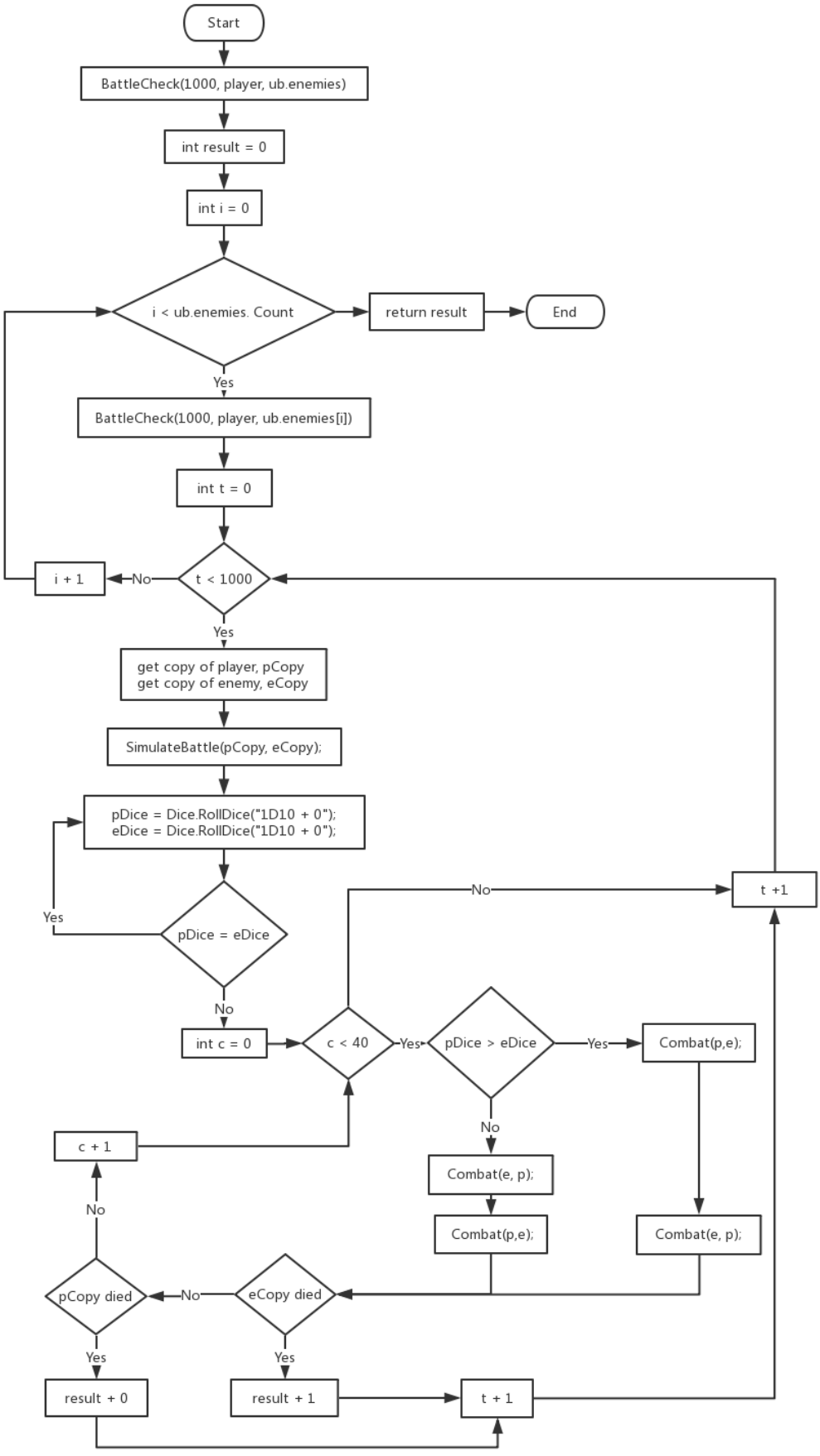


Figure VI: Combat Test process

168 Combinations Test Result

Rank	Class + Race	E	SI	C	Total
1	Barbarian + Dwarf: Hill Dwarf	1952.57	2549.31	5305.00	9806.88
2	Bard + Dwarf: Hill Dwarf	1910.68	2710.59	4751.00	9372.27
3	Cleric + Dwarf: Hill Dwarf	2489.35	3057.40	4868.00	10414.76
4	Druid + Dwarf: Hill Dwarf	2496.08	3056.45	4454.00	10006.53
5	Fighter + Dwarf: Hill Dwarf	1949.75	2557.85	5962.00	10469.60
6	Monk + Dwarf: Hill Dwarf	1966.27	2629.41	4303.00	8898.68
7	Paladin + Dwarf: Hill Dwarf	1955.44	2553.05	5610.00	10118.49
8	Ranger + Dwarf: Hill Dwarf	1955.59	2608.74	5440.00	10004.33
9	Rogue + Dwarf: Hill Dwarf	1964.38	2620.63	5260.00	9845.01
10	Sorcerer + Dwarf: Hill Dwarf	1912.36	2713.12	4184.00	8809.48
11	Warlock + Dwarf: Hill Dwarf	1914.89	2714.72	4695.00	9324.61
12	Wizard + Dwarf: Hill Dwarf	1968.23	2590.73	4176.00	8734.97
13	Barbarian + Dwarf: Mountain Dwarf	2416.18	2971.33	5212.00	10599.51
14	Bard + Dwarf: Mountain Dwarf	2283.65	3162.81	3501.00	8947.46
15	Cleric + Dwarf: Mountain Dwarf	2397.02	3105.62	3506.00	9008.65
16	Druid + Dwarf: Mountain Dwarf	2386.06	3105.80	3336.00	8827.86
17	Fighter + Dwarf: Mountain Dwarf	2421.79	2966.05	6074.00	11461.84
18	Monk + Dwarf: Mountain Dwarf	2336.01	3073.32	3223.00	8632.33
19	Paladin + Dwarf: Mountain Dwarf	2420.33	2973.00	5570.00	10963.33
20	Ranger + Dwarf: Mountain Dwarf	2332.74	3066.32	3346.00	8745.06
21	Rogue + Dwarf: Mountain Dwarf	2343.32	3076.74	5297.00	10717.05
22	Sorcerer + Dwarf: Mountain Dwarf	2286.53	3165.11	3158.00	8609.64
23	Warlock + Dwarf: Mountain Dwarf	2286.50	3168.85	3446.00	8901.35
24	Wizard + Dwarf: Mountain Dwarf	2347.17	3044.03	3019.00	8410.20
25	Barbarian + Elf: High Elf	1645.14	2317.24	5121.00	9083.38
26	Bard + Elf: High Elf	1605.88	2477.15	4288.00	8371.03
27	Cleric + Elf: High Elf	1718.06	2411.13	4458.00	8587.19
28	Druid + Elf: High Elf	1716.91	2408.40	4251.00	8376.32
29	Fighter + Elf: High Elf	1647.53	2313.91	5251.00	9212.44
30	Monk + Elf: High Elf	1863.11	2652.69	4331.00	8846.80
31	Paladin + Elf: High Elf	1642.74	2334.73	5376.00	9353.47
32	Ranger + Elf: High Elf	1862.86	2656.82	5478.00	9997.68
33	Rogue + Elf: High Elf	1866.42	2664.10	5222.00	9752.52
34	Sorcerer + Elf: High Elf	1598.37	2475.82	4076.00	8150.19
35	Warlock + Elf: High Elf	1606.87	2471.82	4211.00	8289.70
36	Wizard + Elf: High Elf	2294.41	2780.96	4108.00	9183.37
37	Barbarian + Elf: Wodd Elf	1744.20	2286.19	5084.00	9114.39
38	Bard + Elf: Wodd Elf	1707.07	2446.35	4279.00	8432.42
39	Cleric + Elf: Wodd Elf	2283.73	2791.79	4500.00	9575.52
40	Druid + Elf: Wodd Elf	2286.19	2785.76	4246.00	9317.94
41	Fighter + Elf: Wodd Elf	1739.89	2279.53	5276.00	9295.42
42	Monk + Elf: Wodd Elf	1903.02	2583.45	4350.00	8836.47
43	Paladin + Elf: Wodd Elf	1739.93	2280.44	5310.00	9330.38
44	Ranger + Elf: Wodd Elf	1896.28	2581.28	5403.00	9880.56
45	Rogue + Elf: Wodd Elf	1904.88	2573.06	5289.00	9766.93
46	Sorcerer + Elf: Wodd Elf	1704.48	2443.67	4046.00	8194.15
47	Warlock + Elf: Wodd Elf	1708.96	2444.21	4234.00	8387.17
48	Wizard + Elf: Wodd Elf	1763.41	2323.92	4090.00	8177.33
49	Barbarian + Elf: Dark Elf	2271.85	1986.13	5067.00	9324.98
50	Bard + Elf: Dark Elf	2171.84	2855.29	4278.00	9305.13
51	Cleric + Elf: Dark Elf	2343.42	2072.61	4456.00	8872.03
52	Druid + Elf: Dark Elf	2342.97	2072.97	4225.00	8640.94
53	Fighter + Elf: Dark Elf	2267.28	1984.16	5289.00	9540.44
54	Monk + Elf: Dark Elf	2426.79	2280.06	4342.00	9048.85
55	Paladin + Elf: Dark Elf	2273.07	1983.15	5402.00	9658.23
56	Ranger + Elf: Dark Elf	2432.13	2279.16	5470.00	10181.29
57	Rogue + Elf: Dark Elf	2428.03	2275.47	5186.00	9889.50
58	Sorcerer + Elf: Dark Elf	2170.37	2860.38	4090.00	9120.75
59	Warlock + Elf: Dark Elf	2164.29	2867.08	4247.00	9278.37
60	Wizard + Elf: Dark Elf	2282.57	2023.91	4058.00	8364.48
61	Barbarian + Halfling: Lightfoot	3172.25	2843.99	5556.00	11572.23
62	Bard + Halfling: Lightfoot	3043.42	3981.03	4385.00	11409.45
63	Cleric + Halfling: Lightfoot	3253.43	2951.03	4681.00	10885.46
64	Druid + Halfling: Lightfoot	3260.86	2952.93	4329.00	10542.79
65	Fighter + Halfling: Lightfoot	3182.59	2844.22	5615.00	11641.81
66	Monk + Halfling: Lightfoot	3379.74	3220.76	4463.00	11063.49
67	Paladin + Halfling: Lightfoot	3175.05	2839.12	5790.00	11804.17
68	Ranger + Halfling: Lightfoot	3383.82	3215.52	5952.00	12551.34
69	Rogue + Halfling: Lightfoot	3381.96	3212.31	5712.00	12306.27
70	Sorcerer + Halfling: Lightfoot	3054.69	3974.93	4213.00	11242.62
71	Warlock + Halfling: Lightfoot	3048.94	3978.42	4324.00	11351.36
72	Wizard + Halfling: Lightfoot	3194.26	2885.08	4221.00	10300.34
73	Barbarian + Halfling: Stout	3173.72	3830.25	5562.00	12565.97
74	Bard + Halfling: Stout	3130.73	4031.40	4420.00	11582.13
75	Cleric + Halfling: Stout	3252.81	3949.36	4652.00	11854.17
76	Druid + Halfling: Stout	3252.39	3955.15	4356.00	11563.54
77	Fighter + Halfling: Stout	3167.61	3845.54	5746.00	12759.15
78	Monk + Halfling: Stout	3383.13	4216.49	4480.00	12079.63
79	Paladin + Halfling: Stout	3173.55	3834.48	5778.00	12786.03
80	Ranger + Halfling: Stout	3380.83	4219.02	5947.00	13546.85
81	Rogue + Halfling: Stout	3384.72	4216.57	5757.00	13358.29
82	Sorcerer + Halfling: Stout	3126.34	4030.45	4199.00	11355.80
83	Warlock + Halfling: Stout	3127.62	4026.13	4350.00	11503.75
84	Wizard + Halfling: Stout	3193.90	3880.46	4238.00	11312.35
85	Barbarian + Human	1080.25	1057.77	5017.00	7155.01
86	Bard + Human	853.60	1977.27	2105.00	4935.87
87	Cleric + Human	1491.04	1606.68	2316.00	5413.72
88	Druid + Human	1492.75	1608.20	2031.00	5131.95
89	Fighter + Human	1076.36	1061.62	5111.00	7248.98
90	Monk + Human	1147.88	1447.51	2926.00	5521.39
91	Paladin + Human	1077.24	1057.96	5267.00	7402.20
92	Ranger + Human	1145.18	1445.68	3111.00	5701.87
93	Rogue + Human	1147.55	1447.96	5287.00	7882.51
94	Sorcerer + Human	851.84	1976.49	1855.00	4683.33
95	Warlock + Human	848.27	1969.78	2004.00	4822.05
96	Wizard + Human	1533.08	1517.11	1562.00	4612.19
97	Barbarian + Dragonborn	2414.69	2222.87	5258.00	9895.56
98	Bard + Dragonborn	2224.04	3124.21	3488.00	8836.25
99	Cleric + Dragonborn	2393.27	2345.48	3494.00	8232.75
100	Druid + Dragonborn	2382.53	2349.93	3269.00	8001.46
101	Fighter + Dragonborn	2420.47	2205.20	6014.00	10639.67
102	Monk + Dragonborn	2336.04	2311.93	3244.00	7891.96
103	Paladin + Dragonborn	2420.92	2218.97	5575.00	10214.89
104	Ranger + Dragonborn	2328.77	2322.95	3389.00	8040.72
105	Rogue + Dragonborn	2330.28	2323.71	5299.00	9952.99
106	Sorcerer + Dragonborn	2225.16	3121.06	3098.00	8444.22
107	Warlock + Dragonborn	2215.20	3124.55	3429.00	8768.74
108	Wizard + Dragonborn	2342.61	2295.18	3036.00	7673.79
109	Barbarian + Gnome: Forest Gnome	1698.86	2316.28	5039.00	9054.15
110	Bard + Gnome: Forest Gnome	1658.01	2491.42	4179.00	8328.43
111	Cleric + Gnome: Forest Gnome	1769.14	2428.42	4381.00	8578.57
112	Druid + Gnome: Forest Gnome	1766.11	2425.15	4085.00	8276.26
113	Fighter + Gnome: Forest Gnome	1703.53	2328.38	5213.00	9244.91
114	Monk + Gnome: Forest Gnome	1952.69	2717.08	4327.00	8996.77
115	Paladin + Gnome: Forest Gnome	1699.00	2326.75	5212.00	9237.75
116	Ranger + Gnome: Forest Gnome	1958.98	2707.54	5455.00	10121.52
117	Rogue + Gnome: Forest Gnome	1955.31	2719.35	5282.00	9956.66
118	Sorcerer + Gnome: Forest Gnome	1655.61	2490.63	3998.00	8144.24
119	Warlock + Gnome: Forest Gnome	1653.17	2480.65	4132.00	8265.82
120	Wizard + Gnome: Forest Gnome	2252.93	2722.19	3993.00	8968.12
121	Barbarian + Gnome: Rock Gnome	1944.32	2645.68	5249.00	9838.99
122	Bard + Gnome: Rock Gnome	1905.02	2802.77	4730.00	9437.79
123	Cleric + Gnome: Rock Gnome	2011.49	2745.82	4862.00	9619.31
124	Druid + Gnome: Rock Gnome	2014.14	2740.81	4478.00	9232.94
125	Fighter + Gnome: Rock Gnome	1940.35	2654.97	6012.00	10607.32
126	Monk + Gnome: Rock Gnome	1951.28	2714.75	4311.00	8977.03
127	Paladin + Gnome: Rock Gnome	1944.83	2655.05	5550.00	10149.87
128	Ranger + Gnome: Rock Gnome	1952.32	2709.32	5367.00	10028.64
129	Rogue + Gnome: Rock Gnome	1951.15	2715.47	5336.00	10002.62
130	Sorcerer + Gnome: Rock Gnome	1907.33	2808.33	4217.00	8932.66
131	Warlock + Gnome: Rock Gnome	1908.13	2795.01	4707.00	9410.13
132	Wizard + Gnome: Rock Gnome	2500.48	3044.49	4196.00	9740.97
133	Barbarian + Half-Elf	2046.09	2008.63	5132.00	9186.72
134	Bard + Half-Elf	1910.60	2791.55	3600.00	8302.16
135	Cleric + Half-Elf	2025.10	2113.23	3667.00	7805.33
136	Druid + Half-Elf	2039.92	2123.36	3475.00	7638.28
137	Fighter + Half-Elf	2029.77	2013.94	5638.00	9681.71
138	Monk + Half-Elf	2085.20	2222.58	3650.00	7957.78
139	Paladin + Half-Elf	2044.57	2005.16	5425.00	9474.73
140	Ranger + Half-Elf	2096.88	2219.17	4250.00	8566.05
141	Rogue + Half-Elf	2069.97	2243.89	5296.00	9609.86
142	Sorcerer + Half-Elf	1905.68	2784.46	3222.00	7912.14
143	Warlock + Half-Elf	1916.82	2780.18	3465.00	8161.99
144	Wizard + Half-Elf	2289.31	2248.53	3099.00	7636.85
145	Barbarian + Half-Orc	2418.49	2969.06	7477.00	12864.55
146	Bard + Half-Orc	2279.52	3162.13	5180.00	10621.65
147	Cleric + Half-Orc	2387.35	3104.71	5084.00	10576.05
148	Druid + Half-Orc	2385.66	3105.27	4745.00	10235.93
149	Fighter + Half-Orc	2417.44	2968.06	8825.00	14210.51
150	Monk + Half-Orc	2329.80	3064.73	4597.00	9991.53
151	Paladin + Half-Orc	2423.68	2967.24	8115.00	13505.92
152	Ranger + Half-Orc	2331.35	3082.11	5673.00	11086.46
153	Rogue + Half-Orc	2326.79	3069.32	7798.00	13194.11
154	Sorcerer + Half-Orc	2283.81	3167.77	4482.00	9933.58
155	Warlock + Half-Orc	2279.36	3166.73	5082.00	10528.09
156	Wizard + Half-Orc	2340.03	3050.94	4406.00	9796.97
157	Barbarian + Tiefling	1846.08	1947.39	5286.00	9079.47
158	Bard + Tiefling	1818.95	2748.21	4714.00	9281.17
159	Cleric + Tiefling	1928.69	2044.97	4863.00	8836.66
160	Druid + Tiefling	1921.83	2052.50	4521.00	8495.33
161	Fighter + Tiefling	1845.97	1960.50	5979.00	9785.47
162	Monk + Tiefling	1861.41	2021.48	4361.00	8243.89
163	Paladin + Tiefling	1853.27	1957.66	5571.00	9381.93
164	Ranger + Tiefling	1862.33	2021.35	5425.00	9308.68
165	Rogue + Tiefling	1862.51	2024.97	5253.00	9140.48
166	Sorcerer + Tiefling	1817.65	2752.17	4206.00	8775.83
167	Warlock + Tiefling	1812.05	2752.54	4682.00	9246.59
168	Wizard + Tiefling	2492.06	2411.08	4185.00	9088.15

Table XIV: All Combination Test Result

Total					
Rank	Class + Race	Value	Rank	Class + Race	Value
1	Fighter + Half-Orc	14210.51	85	Fighter + Elf: Wodd Elf	9295.416
2	Ranger + Halfling: Stout	13546.85	86	Bard + Tiefling	9281.167
3	Paladin + Half-Orc	13505.92	87	Warlock + Elf: Dark Elf	9278.366
4	Rogue + Halfling: Stout	13358.29	88	Warlock + Tiefling	9246.587
5	Rogue + Half-Orc	13194.11	89	Fighter + Gnome: Forest Gnome	9244.912
6	Barbarian + Half-Orc	12864.55	90	Paladin + Gnome: Forest Gnome	9237.753
7	Paladin + Halfling: Stout	12786.03	91	Druid + Gnome: Rock Gnome	9232.942
8	Fighter + Halfling: Stout	12759.15	92	Fighter + Elf: High Elf	9212.442
9	Barbarian + Halfling: Stout	12565.97	93	Barbarian + Half-Elf	9186.721
10	Ranger + Halfling: Lightfoot	12551.34	94	Wizard + Elf: High Elf	9183.369
11	Rogue + Halfling: Lightfoot	12306.27	95	Rogue + Tiefling	9140.479
12	Monk + Halfling: Stout	12079.63	96	Sorcerer + Elf: Dark Elf	9120.747
13	Cleric + Halfling: Stout	11854.17	97	Barbarian + Elf: Wodd Elf	9114.39
14	Paladin + Halfling: Lightfoot	11804.17	98	Wizard + Tiefling	9088.147
15	Fighter + Halfling: Lightfoot	11641.81	99	Barbarian + Elf: High Elf	9083.379
16	Bard + Halfling: Stout	11582.13	100	Barbarian + Tiefling	9079.473
17	Barbarian + Halfling: Lightfoot	11572.23	101	Barbarian + Gnome: Forest Gnome	9054.145
18	Druid + Halfling: Stout	11563.54	102	Monk + Elf: Dark Elf	9048.848
19	Warlock + Halfling: Stout	11503.75	103	Cleric + Dwarf: Mountain Dwarf	9008.648
20	Fighter + Dwarf: Mountain Dwarf	11461.84	104	Monk + Gnome: Forest Gnome	8996.773
21	Bard + Halfling: Lightfoot	11409.45	105	Monk + Gnome: Rock Gnome	8977.029
22	Sorcerer + Halfling: Stout	11355.8	106	Wizard + Gnome: Forest Gnome	8968.121
23	Warlock + Halfling: Lightfoot	11351.36	107	Bard + Dwarf: Mountain Dwarf	8947.459
24	Wizard + Halfling: Stout	11312.35	108	Sorcerer + Gnome: Rock Gnome	8932.664
25	Sorcerer + Halfling: Lightfoot	11242.62	109	Warlock + Dwarf: Mountain Dwarf	8901.351
26	Ranger + Half-Orc	11086.46	110	Monk + Dwarf: Hill Dwarf	8898.676
27	Monk + Halfling: Lightfoot	11063.49	111	Cleric + Elf: Dark Elf	8872.027
28	Paladin + Dwarf: Mountain Dwarf	10963.33	112	Monk + Elf: High Elf	8846.801
29	Cleric + Halfling: Lightfoot	10885.46	113	Cleric + Tiefling	8836.661
30	Rogue + Dwarf: Mountain Dwarf	10717.05	114	Monk + Elf: Wodd Elf	8836.466
31	Fighter + Dragonborn	10639.67	115	Bard + Dragonborn	8836.25
32	Bard + Half-Orc	10621.65	116	Druid + Dwarf: Mountain Dwarf	8827.858
33	Fighter + Gnome: Rock Gnome	10607.32	117	Sorcerer + Dwarf: Hill Dwarf	8809.478
34	Barbarian + Dwarf: Mountain Dwarf	10599.51	118	Sorcerer + Tiefling	8775.826
35	Cleric + Half-Orc	10576.05	119	Warlock + Dragonborn	8768.744
36	Druid + Halfling: Lightfoot	10542.79	120	Ranger + Dwarf: Mountain Dwarf	8745.06
37	Warlock + Half-Orc	10528.09	121	Wizard + Dwarf: Hill Dwarf	8734.968
38	Fighter + Dwarf: Hill Dwarf	10469.6	122	Druid + Elf: Dark Elf	8640.936
39	Cleric + Dwarf: Hill Dwarf	10414.76	123	Monk + Dwarf: Mountain Dwarf	8632.328
40	Wizard + Halfling: Lightfoot	10300.34	124	Sorcerer + Dwarf: Mountain Dwarf	8609.639
41	Druid + Half-Orc	10235.93	125	Cleric + Elf: High Elf	8587.19
42	Paladin + Dragonborn	10214.89	126	Cleric + Gnome: Forest Gnome	8578.567
43	Ranger + Elf: Dark Elf	10181.29	127	Ranger + Half-Elf	8566.054
44	Paladin + Gnome: Rock Gnome	10149.87	128	Druid + Tiefling	8495.329
45	Ranger + Gnome: Forest Gnome	10121.52	129	Sorcerer + Dragonborn	8444.217
46	Paladin + Dwarf: Hill Dwarf	10118.49	130	Bard + Elf: Wodd Elf	8432.419
47	Ranger + Gnome: Rock Gnome	10028.64	131	Wizard + Dwarf: Mountain Dwarf	8410.199
48	Druid + Dwarf: Hill Dwarf	10006.53	132	Warlock + Elf: Wodd Elf	8387.165
49	Ranger + Dwarf: Hill Dwarf	10004.33	133	Druid + Elf: High Elf	8376.315
50	Rogue + Gnome: Rock Gnome	10002.62	134	Bard + Elf: High Elf	8371.026
51	Ranger + Elf: High Elf	9997.677	135	Wizard + Elf: Dark Elf	8364.477
52	Monk + Half-Orc	9991.533	136	Bard + Gnome: Forest Gnome	8328.431
53	Rogue + Gnome: Forest Gnome	9956.663	137	Bard + Half-Elf	8302.158
54	Rogue + Dragonborn	9952.994	138	Warlock + Elf: High Elf	8289.697
55	Sorcerer + Half-Orc	9933.58	139	Druid + Gnome: Forest Gnome	8276.257
56	Barbarian + Dragonborn	9895.56	140	Warlock + Gnome: Forest Gnome	8265.819
57	Rogue + Elf: Dark Elf	9889.5	141	Monk + Tiefling	8243.89
58	Ranger + Elf: Wodd Elf	9880.564	142	Cleric + Dragonborn	8232.752
59	Rogue + Dwarf: Hill Dwarf	9845.005	143	Sorcerer + Elf: Wodd Elf	8194.152
60	Barbarian + Gnome: Rock Gnome	9838.993	144	Wizard + Elf: Wodd Elf	8177.327
61	Barbarian + Dwarf: Hill Dwarf	9806.876	145	Warlock + Half-Elf	8161.991
62	Wizard + Half-Orc	9796.967	146	Sorcerer + Elf: High Elf	8150.195
63	Fighter + Tiefling	9785.474	147	Sorcerer + Gnome: Forest Gnome	8144.235
64	Rogue + Elf: Wodd Elf	9766.932	148	Ranger + Dragonborn	8040.721
65	Rogue + Elf: High Elf	9752.523	149	Druid + Dragonborn	8001.46
66	Wizard + Gnome: Rock Gnome	9740.973	150	Monk + Half-Elf	7957.781
67	Fighter + Half-Elf	9681.709	151	Sorcerer + Half-Elf	7912.136
68	Paladin + Elf: Dark Elf	9658.227	152	Monk + Dragonborn	7891.964
69	Cleric + Gnome: Rock Gnome	9619.315	153	Rogue + Human	7882.511
70	Rogue + Half-Elf	9609.857	154	Cleric + Half-Elf	7805.328
71	Cleric + Elf: Wodd Elf	9575.523	155	Wizard + Dragonborn	7673.79
72	Fighter + Elf: Dark Elf	9540.436	156	Druid + Half-Elf	7638.281
73	Paladin + Half-Elf	9474.727	157	Wizard + Half-Elf	7636.846
74	Bard + Gnome: Rock Gnome	9437.791	158	Paladin + Human	7402.199
75	Warlock + Gnome: Rock Gnome	9410.134	159	Fighter + Human	7248.98
76	Paladin + Tiefling	9381.926	160	Barbarian + Human	7155.014
77	Bard + Dwarf: Hill Dwarf	9372.271	161	Ranger + Human	5701.867
78	Paladin + Elf: High Elf	9353.467	162	Monk + Human	5521.386
79	Paladin + Elf: Wodd Elf	9330.377	163	Cleric + Human	5413.715
80	Barbarian + Elf: Dark Elf	9324.978	164	Druid + Human	5131.948
81	Warlock + Dwarf: Hill Dwarf	9324.61	165	Bard + Human	4935.87
82	Druid + Elf: Wodd Elf	9317.943	166	Warlock + Human	4822.046
83	Ranger + Tiefling	9308.68	167	Sorcerer + Human	4683.329
84	Bard + Elf: Dark Elf	9305.131	168	Wizard + Human	4612.186

Table XV: Total Test Score highest to lowest

Exploration					
Rank	Class + Race	Value	Rank	Class + Race	Value
1	Rogue + Halfling: Stout	3384.72	85	Barbarian + Half-Elf	2046.086
2	Ranger + Halfling: Lightfoot	3383.816	86	Paladin + Half-Elf	2044.565
3	Monk + Halfling: Stout	3383.133	87	Druid + Half-Elf	2039.916
4	Rogue + Halfling: Lightfoot	3381.964	88	Fighter + Half-Elf	2029.772
5	Ranger + Halfling: Stout	3380.829	89	Cleric + Half-Elf	2025.103
6	Monk + Halfling: Lightfoot	3379.739	90	Druid + Gnome: Rock Gnome	2014.135
7	Druid + Halfling: Lightfoot	3260.864	91	Cleric + Gnome: Rock Gnome	2011.495
8	Cleric + Halfling: Lightfoot	3253.429	92	Wizard + Dwarf: Hill Dwarf	1968.235
9	Cleric + Halfling: Stout	3252.812	93	Monk + Dwarf: Hill Dwarf	1966.269
10	Druid + Halfling: Stout	3252.39	94	Rogue + Dwarf: Hill Dwarf	1964.375
11	Wizard + Halfling: Lightfoot	3194.264	95	Ranger + Gnome: Forest Gnome	1958.976
12	Wizard + Halfling: Stout	3193.895	96	Ranger + Dwarf: Hill Dwarf	1955.594
13	Fighter + Halfling: Lightfoot	3182.591	97	Paladin + Dwarf: Hill Dwarf	1955.436
14	Paladin + Halfling: Lightfoot	3175.051	98	Rogue + Gnome: Forest Gnome	1955.311
15	Barbarian + Halfling: Stout	3173.719	99	Monk + Gnome: Forest Gnome	1952.694
16	Paladin + Halfling: Stout	3173.553	100	Barbarian + Dwarf: Hill Dwarf	1952.57
17	Barbarian + Halfling: Lightfoot	3172.246	101	Ranger + Gnome: Rock Gnome	1952.323
18	Fighter + Halfling: Stout	3167.613	102	Monk + Gnome: Rock Gnome	1951.279
19	Bard + Halfling: Stout	3130.726	103	Rogue + Gnome: Rock Gnome	1951.155
20	Warlock + Halfling: Stout	3127.62	104	Fighter + Dwarf: Hill Dwarf	1949.752
21	Sorcerer + Halfling: Stout	3126.345	105	Paladin + Gnome: Rock Gnome	1944.827
22	Sorcerer + Halfling: Lightfoot	3054.691	106	Barbarian + Gnome: Rock Gnome	1944.317
23	Warlock + Halfling: Lightfoot	3048.945	107	Fighter + Gnome: Rock Gnome	1940.353
24	Bard + Halfling: Lightfoot	3043.418	108	Cleric + Tiefling	1928.695
25	Wizard + Gnome: Rock Gnome	2500.481	109	Druid + Tiefling	1921.827
26	Druid + Dwarf: Hill Dwarf	2496.078	110	Warlock + Half-Elf	1916.816
27	Wizard + Tiefling	2492.065	111	Warlock + Dwarf: Hill Dwarf	1914.894
28	Cleric + Dwarf: Hill Dwarf	2489.35	112	Sorcerer + Dwarf: Hill Dwarf	1912.355
29	Ranger + Elf: Dark Elf	2432.135	113	Bard + Dwarf: Hill Dwarf	1910.681
30	Rogue + Elf: Dark Elf	2428.026	114	Bard + Half-Elf	1910.605
31	Monk + Elf: Dark Elf	2426.79	115	Warlock + Gnome: Rock Gnome	1908.127
32	Paladin + Half-Orc	2423.683	116	Sorcerer + Gnome: Rock Gnome	1907.329
33	Fighter + Dwarf: Mountain Dwarf	2421.791	117	Sorcerer + Half-Elf	1905.675
34	Paladin + Dragonborn	2420.916	118	Bard + Gnome: Rock Gnome	1905.021
35	Fighter + Dragonborn	2420.468	119	Rogue + Elf: Wodd Elf	1904.877
36	Paladin + Dwarf: Mountain Dwarf	2420.334	120	Monk + Elf: Wodd Elf	1903.017
37	Barbarian + Half-Orc	2418.491	121	Ranger + Elf: Wodd Elf	1896.282
38	Fighter + Half-Orc	2417.444	122	Rogue + Elf: High Elf	1866.424
39	Barbarian + Dwarf: Mountain Dwarf	2416.179	123	Monk + Elf: High Elf	1863.112
40	Barbarian + Dragonborn	2414.69	124	Ranger + Elf: High Elf	1862.856
41	Cleric + Dwarf: Mountain Dwarf	2397.024	125	Rogue + Tiefling	1862.509
42	Cleric + Dragonborn	2393.27	126	Ranger + Tiefling	1862.327
43	Cleric + Half-Orc	2387.345	127	Monk + Tiefling	1861.409
44	Druid + Dwarf: Mountain Dwarf	2386.058	128	Paladin + Tiefling	1853.267
45	Druid + Half-Orc	2385.658	129	Barbarian + Tiefling	1846.079
46	Druid + Dragonborn	2382.535	130	Fighter + Tiefling	1845.974
47	Wizard + Dwarf: Mountain Dwarf	2347.166	131	Bard + Tiefling	1818.954
48	Cleric + Elf: Dark Elf	2343.422	132	Sorcerer + Tiefling	1817.652
49	Rogue + Dwarf: Mountain Dwarf	2343.317	133	Warlock + Tiefling	1812.047
50	Druid + Elf: Dark Elf	2342.971	134	Cleric + Gnome: Forest Gnome	1769.144
51	Wizard + Dragonborn	2342.608	135	Druid + Gnome: Forest Gnome	1766.111
52	Wizard + Half-Orc	2340.026	136	Wizard + Elf: Wodd Elf	1763.411
53	Monk + Dragonborn	2336.039	137	Barbarian + Elf: Wodd Elf	1744.198
54	Monk + Dwarf: Mountain Dwarf	2336.01	138	Paladin + Elf: Wodd Elf	1739.934
55	Ranger + Dwarf: Mountain Dwarf	2332.739	139	Fighter + Elf: Wodd Elf	1739.887
56	Ranger + Half-Orc	2331.352	140	Cleric + Elf: High Elf	1718.056
57	Rogue + Dragonborn	2330.281	141	Druid + Elf: High Elf	1716.912
58	Monk + Half-Orc	2329.8	142	Warlock + Elf: Wodd Elf	1708.957
59	Ranger + Dragonborn	2328.77	143	Bard + Elf: Wodd Elf	1707.067
60	Rogue + Half-Orc	2326.787	144	Sorcerer + Elf: Wodd Elf	1704.481
61	Wizard + Elf: High Elf	2294.406	145	Fighter + Gnome: Forest Gnome	1703.531
62	Wizard + Half-Elf	2289.313	146	Paladin + Gnome: Forest Gnome	1699.003
63	Sorcerer + Dwarf: Mountain Dwarf	2286.532	147	Barbarian + Gnome: Forest Gnome	1698.861
64	Warlock + Dwarf: Mountain Dwarf	2286.503	148	Bard + Gnome: Forest Gnome	1658.007
65	Druid + Elf: Wodd Elf	2286.185	149	Sorcerer + Gnome: Forest Gnome	1655.61
66	Sorcerer + Half-Orc	2283.81	150	Warlock + Gnome: Forest Gnome	1653.169
67	Cleric + Elf: Wodd Elf	2283.729	151	Fighter + Elf: High Elf	1647.534
68	Bard + Dwarf: Mountain Dwarf	2283.648	152	Barbarian + Elf: High Elf	1645.137
69	Wizard + Elf: Dark Elf	2282.566	153	Paladin + Elf: High Elf	1642.738
70	Bard + Half-Orc	2279.518	154	Warlock + Elf: High Elf	1606.874
71	Warlock + Half-Orc	2279.36	155	Bard + Elf: High Elf	1605.88
72	Paladin + Elf: Dark Elf	2273.072	156	Sorcerer + Elf: High Elf	1598.374
73	Barbarian + Elf: Dark Elf	2271.85	157	Wizard + Human	1533.077
74	Fighter + Elf: Dark Elf	2267.276	158	Druid + Human	1492.751
75	Wizard + Gnome: Forest Gnome	2252.929	159	Cleric + Human	1491.038
76	Sorcerer + Dragonborn	2225.16	160	Monk + Human	1147.876
77	Bard + Dragonborn	2224.043	161	Rogue + Human	1147.547
78	Warlock + Dragonborn	2215.198	162	Ranger + Human	1145.183
79	Bard + Elf: Dark Elf	2171.841	163	Barbarian + Human	1080.249
80	Sorcerer + Elf: Dark Elf	2170.368	164	Paladin + Human	1077.243
81	Warlock + Elf: Dark Elf	2164.288	165	Fighter + Human	1076.364
82	Ranger + Half-Elf	2096.884	166	Bard + Human	853.5996
83	Monk + Half-Elf	2085.196	167	Sorcerer + Human	851.8406
84	Rogue + Half-Elf	2069.966	168	Warlock + Human	848.2666

Table XVI: Exploration Test Score Rank

Social Interaction					
Rank	Class + Race	Value	Rank	Class + Race	Value
1	Ranger + Halfling: Stout	4219.018	85	Fighter + Gnome: Rock Gnome	2654.965
2	Rogue + Halfling: Stout	4216.575	86	Monk + Elf: High Elf	2652.689
3	Monk + Halfling: Stout	4216.493	87	Barbarian + Gnome: Rock Gnome	2645.677
4	Bard + Halfling: Stout	4031.4	88	Monk + Dwarf: Hill Dwarf	2629.406
5	Sorcerer + Halfling: Stout	4030.453	89	Rogue + Dwarf: Hill Dwarf	2620.63
6	Warlock + Halfling: Stout	4026.128	90	Ranger + Dwarf: Hill Dwarf	2608.736
7	Bard + Halfling: Lightfoot	3981.029	91	Wizard + Dwarf: Hill Dwarf	2590.734
8	Warlock + Halfling: Lightfoot	3978.416	92	Monk + Elf: Wodd Elf	2583.45
9	Sorcerer + Halfling: Lightfoot	3974.931	93	Ranger + Elf: Wodd Elf	2581.282
10	Druid + Halfling: Stout	3955.146	94	Rogue + Elf: Wodd Elf	2573.056
11	Cleric + Halfling: Stout	3949.36	95	Fighter + Dwarf: Hill Dwarf	2557.851
12	Wizard + Halfling: Stout	3880.457	96	Paladin + Dwarf: Hill Dwarf	2553.055
13	Fighter + Halfling: Stout	3845.541	97	Barbarian + Dwarf: Hill Dwarf	2549.306
14	Paladin + Halfling: Stout	3834.479	98	Bard + Gnome: Forest Gnome	2491.423
15	Barbarian + Halfling: Stout	3830.248	99	Sorcerer + Gnome: Forest Gnome	2490.625
16	Monk + Halfling: Lightfoot	3220.755	100	Warlock + Gnome: Forest Gnome	2480.65
17	Ranger + Halfling: Lightfoot	3215.523	101	Bard + Elf: High Elf	2477.147
18	Rogue + Halfling: Lightfoot	3212.31	102	Sorcerer + Elf: High Elf	2475.821
19	Warlock + Dwarf: Mountain Dwarf	3168.848	103	Warlock + Elf: High Elf	2471.823
20	Sorcerer + Half-Orc	3167.77	104	Bard + Elf: Wodd Elf	2446.352
21	Warlock + Half-Orc	3166.729	105	Warlock + Elf: Wodd Elf	2444.208
22	Sorcerer + Dwarf: Mountain Dwarf	3165.107	106	Sorcerer + Elf: Wodd Elf	2443.671
23	Bard + Dwarf: Mountain Dwarf	3162.811	107	Cleric + Gnome: Forest Gnome	2428.423
24	Bard + Half-Orc	3162.132	108	Druid + Gnome: Forest Gnome	2425.147
25	Warlock + Dragonborn	3124.546	109	Cleric + Elf: High Elf	2411.135
26	Bard + Dragonborn	3124.207	110	Wizard + Tiefling	2411.082
27	Sorcerer + Dragonborn	3121.057	111	Druid + Elf: High Elf	2408.403
28	Druid + Dwarf: Mountain Dwarf	3105.8	112	Druid + Dragonborn	2349.926
29	Cleric + Dwarf: Mountain Dwarf	3105.624	113	Cleric + Dragonborn	2345.482
30	Druid + Half-Orc	3105.269	114	Paladin + Elf: High Elf	2334.729
31	Cleric + Half-Orc	3104.707	115	Fighter + Gnome: Forest Gnome	2328.382
32	Ranger + Half-Orc	3082.106	116	Paladin + Gnome: Forest Gnome	2326.75
33	Rogue + Dwarf: Mountain Dwarf	3076.737	117	Wizard + Elf: Wodd Elf	2323.916
34	Monk + Dwarf: Mountain Dwarf	3073.319	118	Rogue + Dragonborn	2323.713
35	Rogue + Half-Orc	3069.321	119	Ranger + Dragonborn	2322.951
36	Ranger + Dwarf: Mountain Dwarf	3066.32	120	Barbarian + Elf: High Elf	2317.242
37	Monk + Half-Orc	3064.732	121	Barbarian + Gnome: Forest Gnome	2316.284
38	Cleric + Dwarf: Hill Dwarf	3057.405	122	Fighter + Elf: High Elf	2313.908
39	Druid + Dwarf: Hill Dwarf	3056.455	123	Monk + Dragonborn	2311.925
40	Wizard + Half-Orc	3050.94	124	Wizard + Dragonborn	2295.183
41	Wizard + Gnome: Rock Gnome	3044.492	125	Barbarian + Elf: Wodd Elf	2286.193
42	Wizard + Dwarf: Mountain Dwarf	3044.033	126	Paladin + Elf: Wodd Elf	2280.443
43	Paladin + Dwarf: Mountain Dwarf	2972.998	127	Monk + Elf: Dark Elf	2280.058
44	Barbarian + Dwarf: Mountain Dwarf	2971.333	128	Fighter + Elf: Wodd Elf	2279.529
45	Barbarian + Half-Orc	2969.057	129	Ranger + Elf: Dark Elf	2279.157
46	Fighter + Half-Orc	2968.065	130	Rogue + Elf: Dark Elf	2275.473
47	Paladin + Half-Orc	2967.239	131	Wizard + Half-Elf	2248.533
48	Fighter + Dwarf: Mountain Dwarf	2966.051	132	Rogue + Half-Elf	2243.891
49	Druid + Halfling: Lightfoot	2952.926	133	Barbarian + Dragonborn	2222.87
50	Cleric + Halfling: Lightfoot	2951.033	134	Monk + Half-Elf	2222.584
51	Wizard + Halfling: Lightfoot	2885.08	135	Ranger + Half-Elf	2219.17
52	Warlock + Elf: Dark Elf	2867.078	136	Paladin + Dragonborn	2218.975
53	Sorcerer + Elf: Dark Elf	2860.379	137	Fighter + Dragonborn	2205.199
54	Bard + Elf: Dark Elf	2855.29	138	Druid + Half-Elf	2123.364
55	Fighter + Halfling: Lightfoot	2844.222	139	Cleric + Half-Elf	2113.225
56	Barbarian + Halfling: Lightfoot	2843.986	140	Druid + Elf: Dark Elf	2072.965
57	Paladin + Halfling: Lightfoot	2839.122	141	Cleric + Elf: Dark Elf	2072.605
58	Sorcerer + Gnome: Rock Gnome	2808.335	142	Druid + Tiefling	2052.502
59	Bard + Gnome: Rock Gnome	2802.771	143	Cleric + Tiefling	2044.967
60	Warlock + Gnome: Rock Gnome	2795.007	144	Rogue + Tiefling	2024.97
61	Cleric + Elf: Wodd Elf	2791.794	145	Wizard + Elf: Dark Elf	2023.911
62	Bard + Half-Elf	2791.553	146	Monk + Tiefling	2021.481
63	Druid + Elf: Wodd Elf	2785.758	147	Ranger + Tiefling	2021.353
64	Sorcerer + Half-Elf	2784.46	148	Fighter + Half-Elf	2013.937
65	Wizard + Elf: High Elf	2780.963	149	Barbarian + Half-Elf	2008.635
66	Warlock + Half-Elf	2780.175	150	Paladin + Half-Elf	2005.162
67	Warlock + Tiefling	2752.54	151	Barbarian + Elf: Dark Elf	1986.128
68	Sorcerer + Tiefling	2752.174	152	Fighter + Elf: Dark Elf	1984.16
69	Bard + Tiefling	2748.213	153	Paladin + Elf: Dark Elf	1983.155
70	Cleric + Gnome: Rock Gnome	2745.82	154	Bard + Human	1977.271
71	Druid + Gnome: Rock Gnome	2740.807	155	Sorcerer + Human	1976.489
72	Wizard + Gnome: Forest Gnome	2722.192	156	Warlock + Human	1969.779
73	Rogue + Gnome: Forest Gnome	2719.352	157	Fighter + Tiefling	1960.499
74	Monk + Gnome: Forest Gnome	2717.079	158	Paladin + Tiefling	1957.659
75	Rogue + Gnome: Rock Gnome	2715.468	159	Barbarian + Tiefling	1947.394
76	Monk + Gnome: Rock Gnome	2714.75	160	Druid + Human	1608.197
77	Warlock + Dwarf: Hill Dwarf	2714.716	161	Cleric + Human	1606.677
78	Sorcerer + Dwarf: Hill Dwarf	2713.122	162	Wizard + Human	1517.11
79	Bard + Dwarf: Hill Dwarf	2710.59	163	Rogue + Human	1447.965
80	Ranger + Gnome: Rock Gnome	2709.316	164	Monk + Human	1447.51
81	Ranger + Gnome: Forest Gnome	2707.541	165	Ranger + Human	1445.684
82	Rogue + Elf: High Elf	2664.099	166	Fighter + Human	1061.616
83	Ranger + Elf: High Elf	2656.821	167	Paladin + Human	1057.956
84	Paladin + Gnome: Rock Gnome	2655.046	168	Barbarian + Human	1057.765

Table XVII: Social Interaction Test Rank

Combat					
Rank	Class + Race	Value	Rank	Class + Race	Value
1	Fighter + Half-Orc	8825.00	85	Sorcerer + Half-Orc	4482.00
2	Paladin + Half-Orc	8115.00	86	Monk + Halfling: Stout	4480.00
3	Rogue + Half-Orc	7798.00	87	Druid + Gnome: Rock Gnome	4478.00
4	Barbarian + Half-Orc	7477.00	88	Monk + Halfling: Lightfoot	4463.00
5	Fighter + Dwarf: Mountain Dwarf	6074.00	89	Cleric + Elf: High Elf	4458.00
6	Fighter + Dragonborn	6014.00	90	Cleric + Elf: Dark Elf	4456.00
7	Fighter + Gnome: Rock Gnome	6012.00	91	Druid + Dwarf: Hill Dwarf	4454.00
8	Fighter + Tiefling	5979.00	92	Bard + Halfling: Stout	4420.00
9	Fighter + Dwarf: Hill Dwarf	5962.00	93	Wizard + Half-Orc	4406.00
10	Ranger + Halfling: Lightfoot	5952.00	94	Bard + Halfling: Lightfoot	4385.00
11	Ranger + Halfling: Stout	5947.00	95	Cleric + Gnome: Forest Gnome	4381.00
12	Paladin + Halfling: Lightfoot	5790.00	96	Monk + Tiefling	4361.00
13	Paladin + Halfling: Stout	5778.00	97	Druid + Halfling: Stout	4356.00
14	Rogue + Halfling: Stout	5757.00	98	Monk + Elf: Wodd Elf	4350.00
15	Fighter + Halfling: Stout	5746.00	99	Warlock + Halfling: Stout	4350.00
16	Rogue + Halfling: Lightfoot	5712.00	100	Monk + Elf: Dark Elf	4342.00
17	Ranger + Half-Orc	5673.00	101	Monk + Elf: High Elf	4331.00
18	Fighter + Half-Elf	5638.00	102	Druid + Halfling: Lightfoot	4329.00
19	Fighter + Halfling: Lightfoot	5615.00	103	Monk + Gnome: Forest Gnome	4327.00
20	Paladin + Dwarf: Hill Dwarf	5610.00	104	Warlock + Halfling: Lightfoot	4324.00
21	Paladin + Dragonborn	5575.00	105	Monk + Gnome: Rock Gnome	4311.00
22	Paladin + Tiefling	5571.00	106	Monk + Dwarf: Hill Dwarf	4303.00
23	Paladin + Dwarf: Mountain Dwarf	5570.00	107	Bard + Elf: High Elf	4288.00
24	Barbarian + Halfling: Stout	5562.00	108	Bard + Elf: Wodd Elf	4279.00
25	Barbarian + Halfling: Lightfoot	5556.00	109	Bard + Elf: Dark Elf	4278.00
26	Paladin + Gnome: Rock Gnome	5550.00	110	Druid + Elf: High Elf	4251.00
27	Ranger + Elf: High Elf	5478.00	111	Ranger + Half-Elf	4250.00
28	Ranger + Elf: Dark Elf	5470.00	112	Warlock + Elf: Dark Elf	4247.00
29	Ranger + Gnome: Forest Gnome	5455.00	113	Druid + Elf: Wodd Elf	4246.00
30	Ranger + Dwarf: Hill Dwarf	5440.00	114	Wizard + Halfling: Stout	4238.00
31	Paladin + Half-Elf	5425.00	115	Warlock + Elf: Wodd Elf	4234.00
32	Ranger + Tiefling	5425.00	116	Druid + Elf: Dark Elf	4225.00
33	Ranger + Elf: Wodd Elf	5403.00	117	Wizard + Halfling: Lightfoot	4221.00
34	Paladin + Elf: Dark Elf	5402.00	118	Sorcerer + Gnome: Rock Gnome	4217.00
35	Paladin + Elf: High Elf	5376.00	119	Sorcerer + Halfling: Lightfoot	4213.00
36	Ranger + Gnome: Rock Gnome	5367.00	120	Warlock + Elf: High Elf	4211.00
37	Rogue + Gnome: Rock Gnome	5336.00	121	Sorcerer + Tiefling	4206.00
38	Paladin + Elf: Wodd Elf	5310.00	122	Sorcerer + Halfling: Stout	4199.00
39	Barbarian + Dwarf: Hill Dwarf	5305.00	123	Wizard + Gnome: Rock Gnome	4196.00
40	Rogue + Dragonborn	5299.00	124	Wizard + Tiefling	4185.00
41	Rogue + Dwarf: Mountain Dwarf	5297.00	125	Sorcerer + Dwarf: Hill Dwarf	4184.00
42	Rogue + Half-Elf	5296.00	126	Bard + Gnome: Forest Gnome	4179.00
43	Rogue + Elf: Wodd Elf	5289.00	127	Wizard + Dwarf: Hill Dwarf	4176.00
44	Fighter + Elf: Dark Elf	5289.00	128	Warlock + Gnome: Forest Gnome	4132.00
45	Rogue + Human	5287.00	129	Wizard + Elf: High Elf	4108.00
46	Barbarian + Tiefling	5286.00	130	Wizard + Elf: Wodd Elf	4090.00
47	Rogue + Gnome: Forest Gnome	5282.00	131	Sorcerer + Elf: Dark Elf	4090.00
48	Fighter + Elf: Wodd Elf	5276.00	132	Druid + Gnome: Forest Gnome	4085.00
49	Paladin + Human	5267.00	133	Sorcerer + Elf: High Elf	4076.00
50	Rogue + Dwarf: Hill Dwarf	5260.00	134	Wizard + Elf: Dark Elf	4058.00
51	Barbarian + Dragonborn	5258.00	135	Sorcerer + Elf: Wodd Elf	4046.00
52	Rogue + Tiefling	5253.00	136	Sorcerer + Gnome: Forest Gnome	3998.00
53	Fighter + Elf: High Elf	5251.00	137	Wizard + Gnome: Forest Gnome	3993.00
54	Barbarian + Gnome: Rock Gnome	5249.00	138	Cleric + Half-Elf	3667.00
55	Rogue + Elf: High Elf	5222.00	139	Monk + Half-Elf	3650.00
56	Fighter + Gnome: Forest Gnome	5213.00	140	Bard + Half-Elf	3600.00
57	Barbarian + Dwarf: Mountain Dwarf	5212.00	141	Cleric + Dwarf: Mountain Dwarf	3506.00
58	Paladin + Gnome: Forest Gnome	5212.00	142	Bard + Dwarf: Mountain Dwarf	3501.00
59	Rogue + Elf: Dark Elf	5186.00	143	Cleric + Dragonborn	3494.00
60	Bard + Half-Orc	5180.00	144	Bard + Dragonborn	3488.00
61	Barbarian + Half-Elf	5132.00	145	Druid + Half-Elf	3475.00
62	Barbarian + Elf: High Elf	5121.00	146	Warlock + Half-Elf	3465.00
63	Fighter + Human	5111.00	147	Warlock + Dwarf: Mountain Dwarf	3446.00
64	Barbarian + Elf: Wodd Elf	5084.00	148	Warlock + Dragonborn	3429.00
65	Cleric + Half-Orc	5084.00	149	Ranger + Dragonborn	3389.00
66	Warlock + Half-Orc	5082.00	150	Ranger + Dwarf: Mountain Dwarf	3346.00
67	Barbarian + Elf: Dark Elf	5067.00	151	Druid + Dwarf: Mountain Dwarf	3336.00
68	Barbarian + Gnome: Forest Gnome	5039.00	152	Druid + Dragonborn	3269.00
69	Barbarian + Human	5017.00	153	Monk + Dragonborn	3244.00
70	Cleric + Dwarf: Hill Dwarf	4868.00	154	Monk + Dwarf: Mountain Dwarf	3223.00
71	Cleric + Tiefling	4863.00	155	Sorcerer + Half-Elf	3222.00
72	Cleric + Gnome: Rock Gnome	4862.00	156	Sorcerer + Dwarf: Mountain Dwarf	3158.00
73	Bard + Dwarf: Hill Dwarf	4751.00	157	Ranger + Human	3111.00
74	Druid + Half-Orc	4745.00	158	Wizard + Half-Elf	3099.00
75	Bard + Gnome: Rock Gnome	4730.00	159	Sorcerer + Dragonborn	3098.00
76	Bard + Tiefling	4714.00	160	Wizard + Dragonborn	3036.00
77	Warlock + Gnome: Rock Gnome	4707.00	161	Wizard + Dwarf: Mountain Dwarf	3019.00
78	Warlock + Dwarf: Hill Dwarf	4695.00	162	Monk + Human	2926.00
79	Warlock + Tiefling	4682.00	163	Cleric + Human	2316.00
80	Cleric + Halfling: Lightfoot	4681.00	164	Bard + Human	2105.00
81	Cleric + Halfling: Stout	4652.00	165	Druid + Human	2031.00
82	Monk + Half-Orc	4597.00	166	Warlock + Human	2004.00
83	Druid + Tiefling	4521.00	167	Sorcerer + Human	1855.00
84	Cleric + Elf: Wodd Elf	4500.00	168	Wizard + Human	1562.00

Table XVIII: Combat Test Score Rank