****

**Programming Principles 01**

**Coursework02**

**Module Leader’s Name – Mr. Guganathan Poravi**

**Name : Shajeeth Suwarnarajah.**

**UOW NO : w1697765.**

**Student Id no : 2017330**.

**Introduction**

The Report is based known board game Dungeons and Dragons. According to the following course work 2, at the first we have to enter the level. Then select the character class. Then we have to select the dice type for the dice roll. Then the dice rolled values will be add to the attributes. we have to calculate the bonus for each attributes. At the end we should calculate the hitpoints according to the specification and display the variables with the values and their bonus and end it with hitpoints.

**Analysis**

**Functional Requirements**

This solution is to get the statistics from user and calculate the required output for the user.

* Getting user inputs for statistics such as,
  + Level
  + Character class
  + Dice type
  + skills
* Calculating the relevant statistics,
  + Character Ability such as Str, Dex, Con, etc.
  + Bonus for each Ability.
  + Hit-Points of the character
  + Proficiency bonus
  + Stat affinity
  + Base Attack Bonus
  + Compact and damage
  + Skill points
  + rank
* Displaying the statistics

**Non-Functional Requirements**

* Validating user inputs
* Allowing the user to select a dice roll type
* Terminating the program if inputs are invalid.
* Formatting the output for a better readability

**Pseudocode**

BEGIN

Initialize level, attributes to zero

Initialize bonus1, bonus2, bonus3, bonus4, bonus5, bonus6, hitpoints min sum to zero

Initialize n to 10

GET level

IF 0 > level > 21 THEN

Display “Invalid Level”

Else

FOR(1 to 4)

Sum = 4 random numbers – minimum of the 4 random numbers

SET finalRoll to dicetype2()

END FOR

FOR(1 to 6)

n=n-1

FOR(0 to n)

roll[j] = ((Math.*random*() \* 1000 % 6 + 1))  
sum += roll[j]

END FOR

finalSum[i]=sum

SET finalSum to dicerollTxMethod()

END FOR

CASE diceRoll

1:

FOR(1 to 6)

attributes[i] = input.nextInt()

IF (attributes[i] > 10)

THEN  
 bonus[i] = (attributes[i] - 10) / 2

ELSE  
bonus[i] = (attributes[i] - 11) / 2

END FOR

2:

FOR(1 to 6)

attributes[i] = dicetype2()

IF (attributes[i] > 10)

THEN  
 bonus[i] = (attributes[i] - 10) / 2

ELSE  
 bonus[i] = (attributes[i] - 11) / 2

END FOR

3:

FOR(1 to 6)

attributes[i] = diceType2 ()

IF (attributes[i] >= 16)

THEN  
 attributes[i] += (int) ((Math.*random*() \* 1000 % 6 + 1))

IF (attributes[i] > 10)

THEN  
 bonus[i] = (attributes[i] - 10) / 2

ELSE  
 bonus[i] = (attributes[i] - 11) / 2

END FOR

4:

FOR(1 to 6)

attributes[i] = dicerollTxMethod()

IF (attributes[i] > 10)

THEN  
 bonus[i] = (attributes[i] - 10) / 2

ELSE  
 bonus[i] = (attributes[i] - 11) / 2

END FOR

ENDCASE

CASE characterClass OF

Barbarian:

hitdice = (int) (Math.*random*() \* 1000 % 12 + 1);  
hitpoints = hitdice + bonus

Bard:

hitdice = (int) (Math.*random*() \* 1000 % 12 + 1);  
hitpoints = hitdice + bonus

Cleric:

hitdice = (int) (Math.*random*() \* 1000 % 12 + 1);  
hitpoints = hitdice + bonus

Druid:

hitdice = (int) (Math.*random*() \* 1000 % 12 + 1);  
hitpoints = hitdice + bonus

Fighter:

hitdice = (int) (Math.*random*() \* 1000 % 12 + 1);  
hitpoints = hitdice + bonus

Monk:

hitdice = (int) (Math.*random*() \* 1000 % 12 + 1);  
hitpoints = hitdice + bonus

Paladin:

hitdice = (int) (Math.*random*() \* 1000 % 12 + 1);  
hitpoints = hitdice + bonus

Ranger:

hitdice = (int) (Math.*random*() \* 1000 % 12 + 1);  
hitpoints = hitdice + bonus

Rougue:

hitdice = (int) (Math.*random*() \* 1000 % 12 + 1);  
hitpoints = hitdice + bonus

Sorcerer:

hitdice = (int) (Math.*random*() \* 1000 % 12 + 1);  
hitpoints = hitdice + bonus

Warlock:

hitdice = (int) (Math.*random*() \* 1000 % 12 + 1);  
hitpoints = hitdice + bonus

Wizard:

hitdice = (int) (Math.*random*() \* 1000 % 12 + 1);  
hitpoints = hitdice + bonus

ENDCASE

CASE(selectedSkillsList)

Acrobatics

rank += 0.5;  
statAffinity += bonus

Animal handling

rank += 0.5;  
statAffinity += bonus

Arcana

rank += 0.5;  
statAffinity += bonus

Athletics

rank += 0.5;  
statAffinity += bonus

Deception

rank += 0.5;  
statAffinity += bonus

History

rank += 0.5;  
statAffinity += bonus

Insight

rank += 0.5;  
statAffinity += bonus

Intimidation

rank += 0.5;  
statAffinity += bonus

Investigation

rank += 0.5;  
statAffinity += bonus

Medicine

rank += 0.5;  
statAffinity += bonus

Nature

rank += 0.5;  
statAffinity += bonus

Percepation

rank += 0.5;  
statAffinity += bonus

Performance

rank += 0.5;  
statAffinity += bonus

Religion

rank += 0.5;  
statAffinity += bonus

Sleight

rank += 0.5;  
statAffinity += bonus

Stealth

rank += 0.5;  
statAffinity += bonus

Persuasion

rank += 0.5;  
statAffinity += bonus

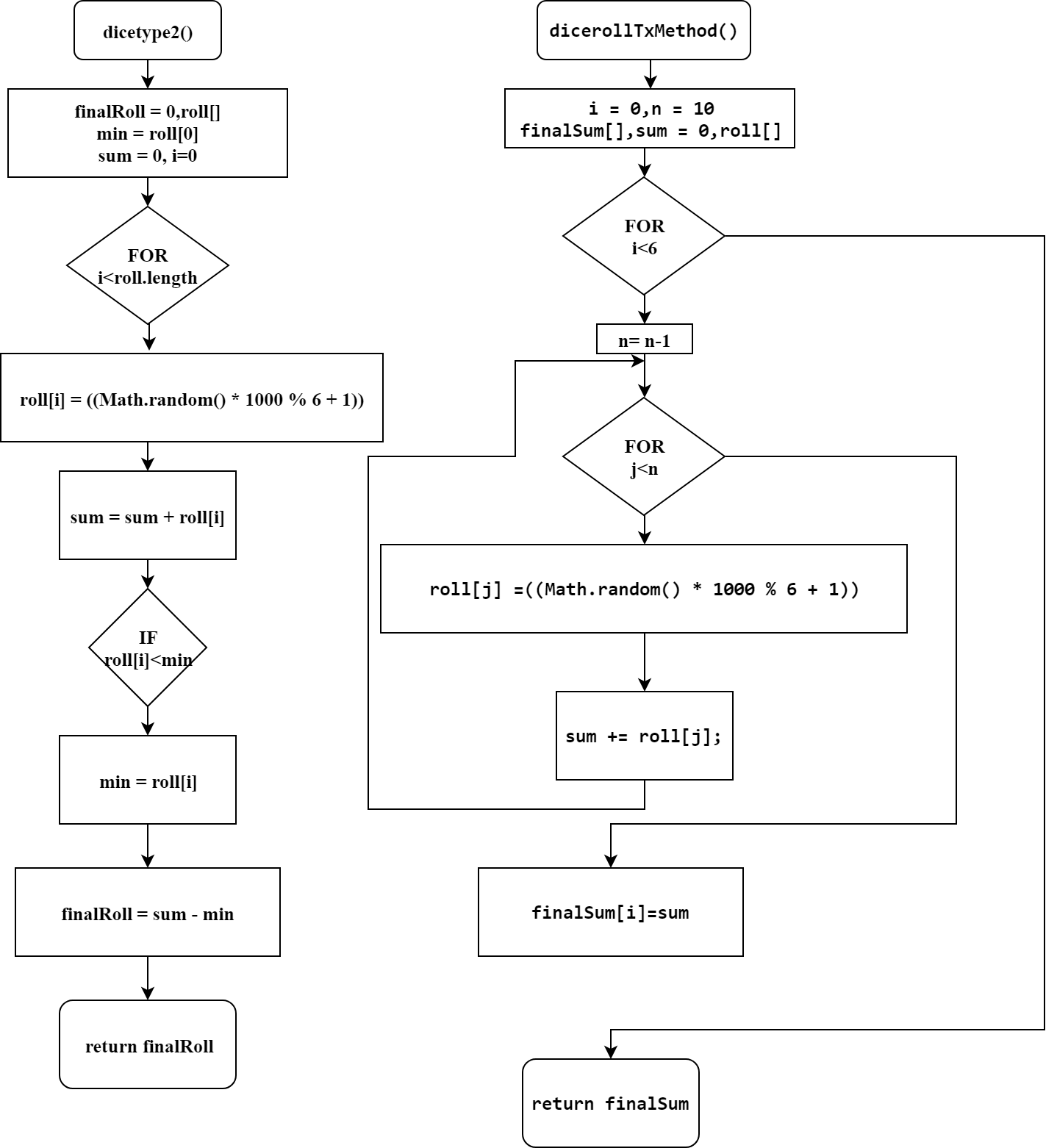
Survial

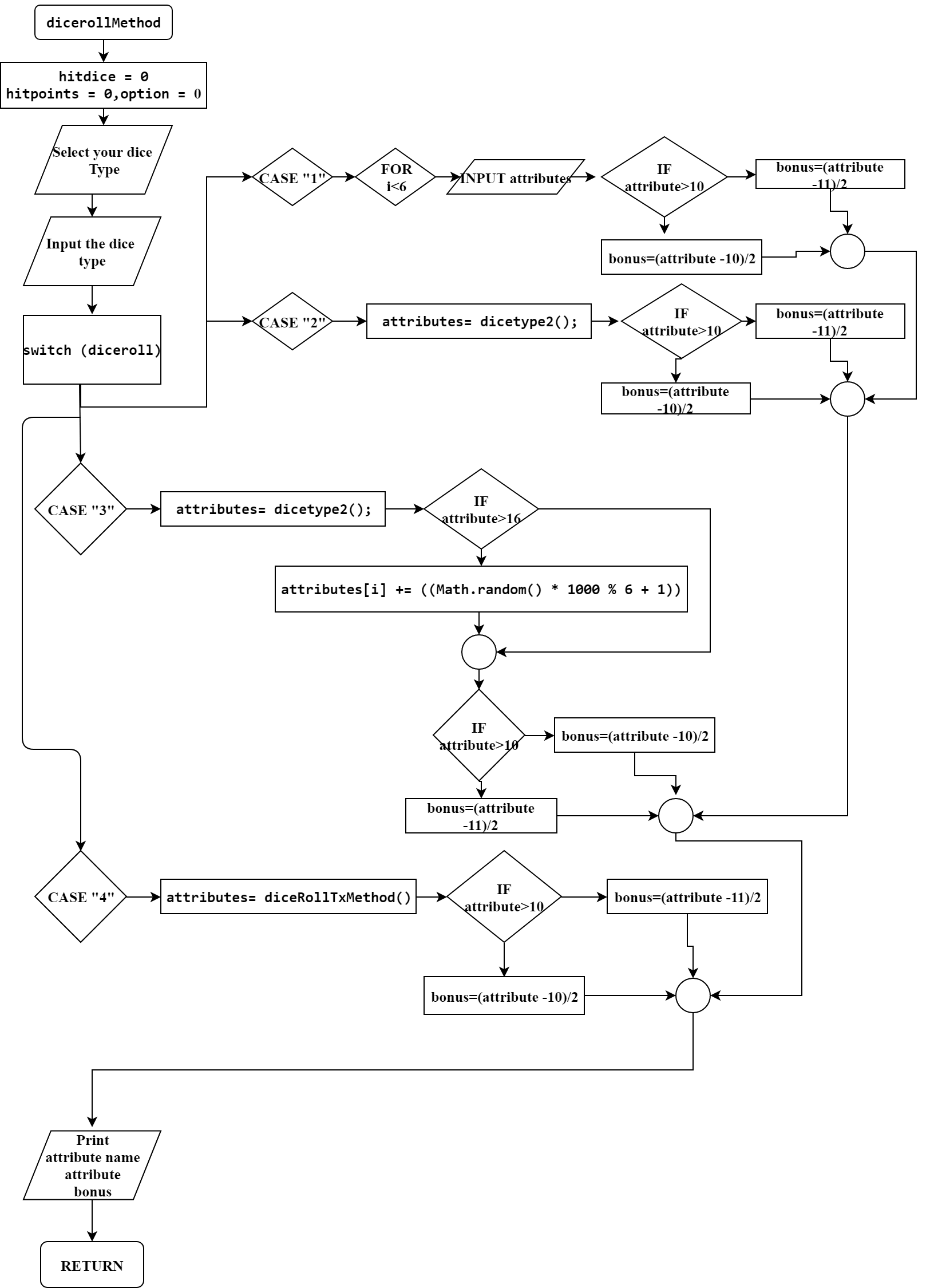
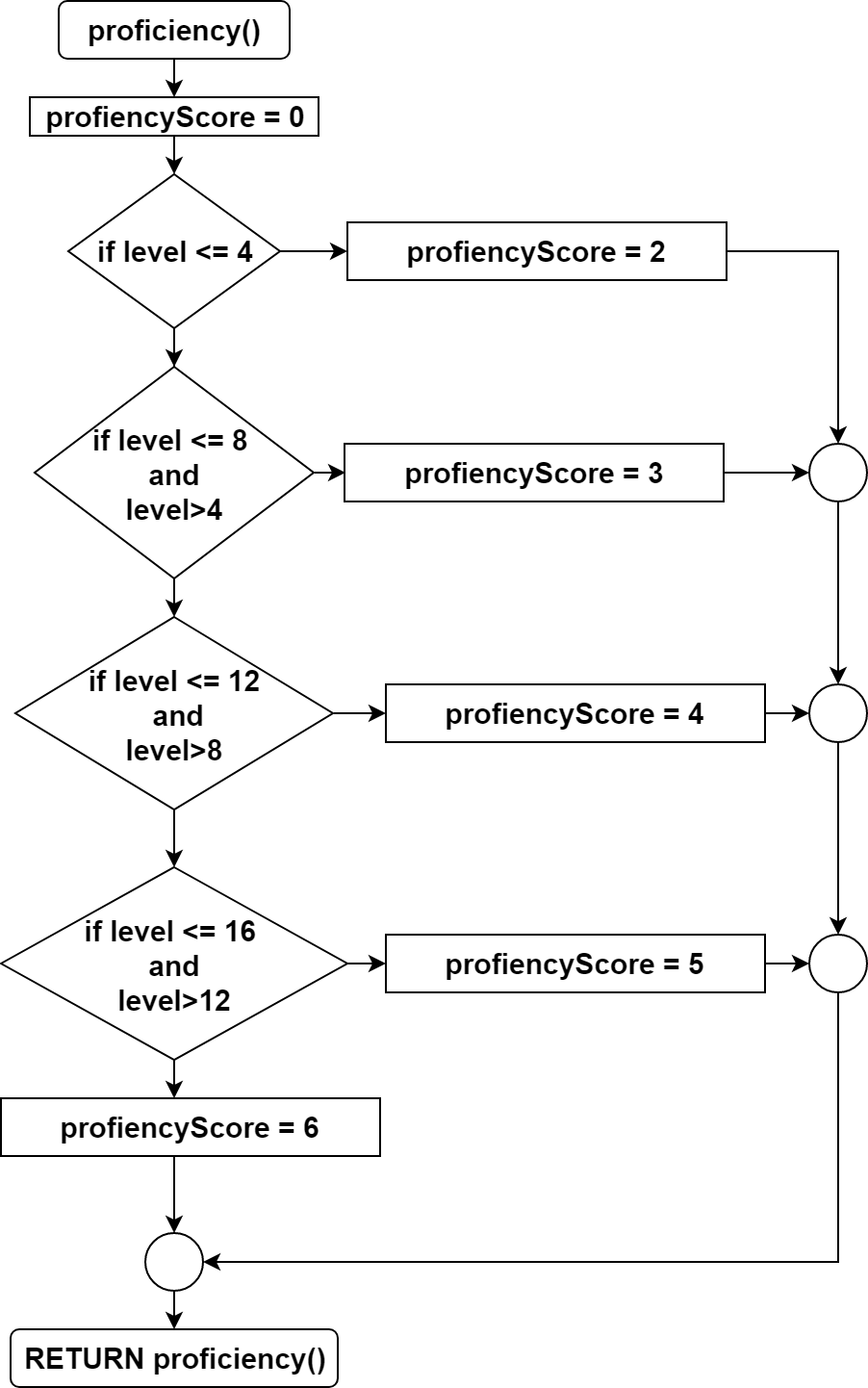
rank += 0.5;  
statAffinity += bonus

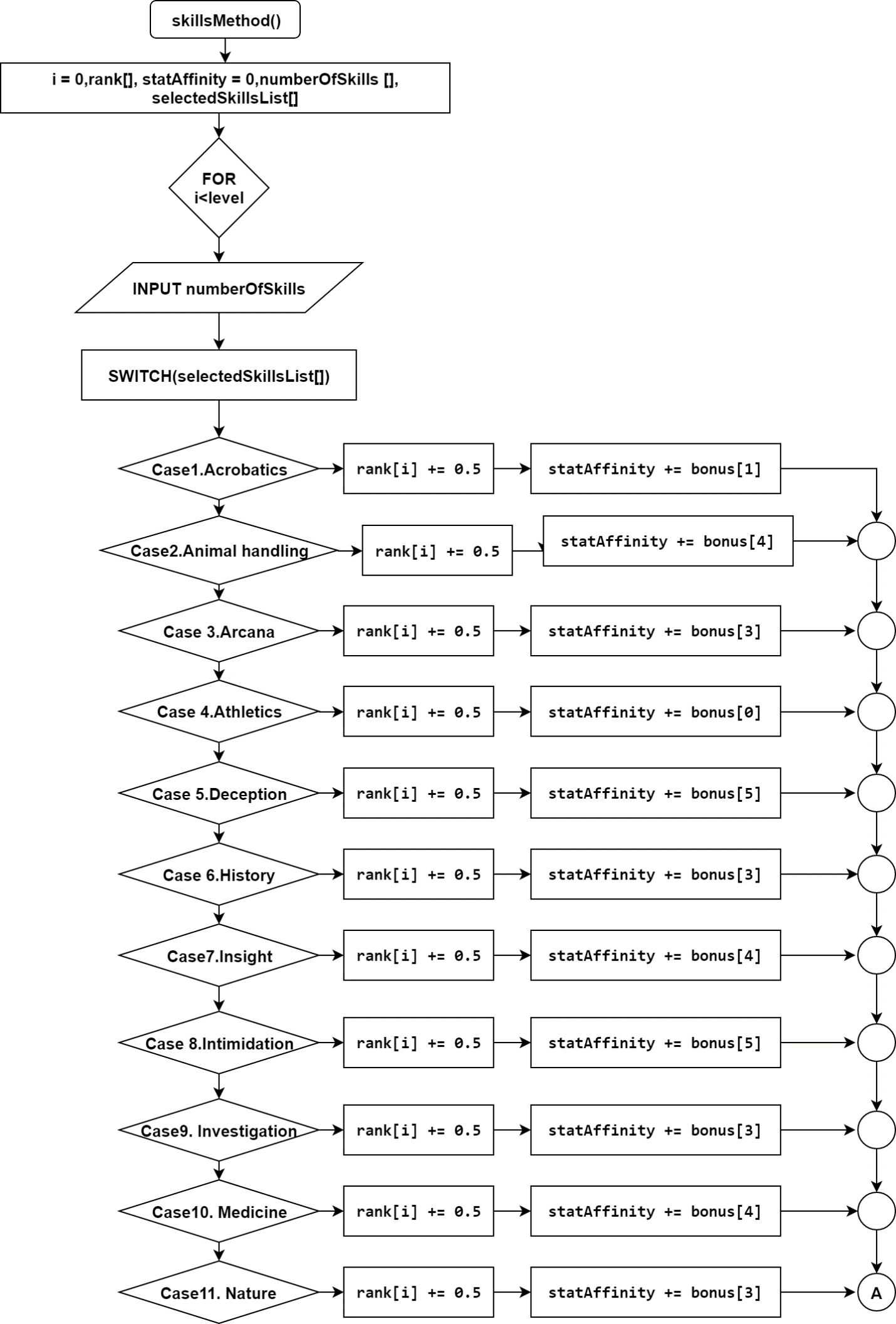
PRINT hitpoints,selectedSkillsList,statAffinity

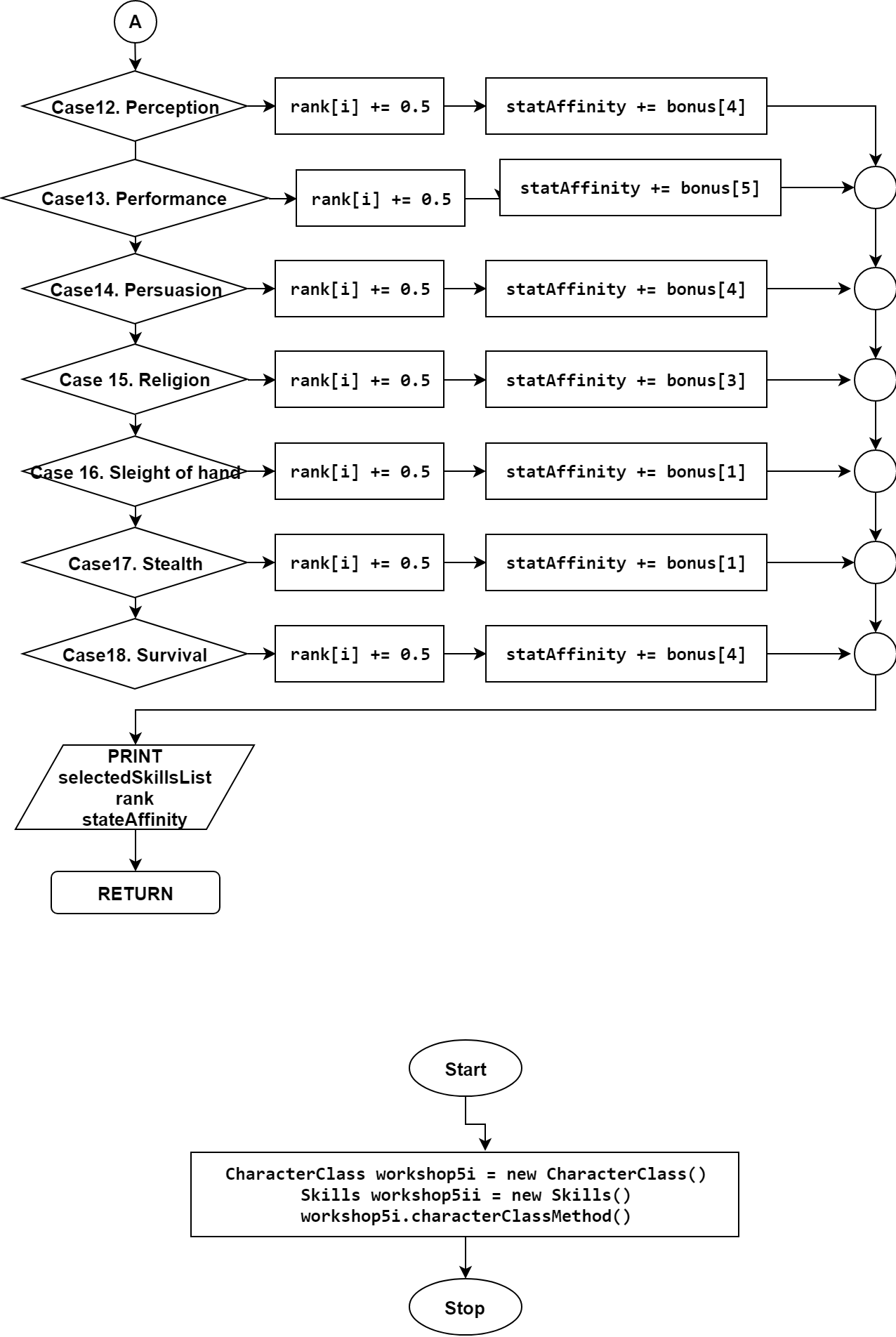
END

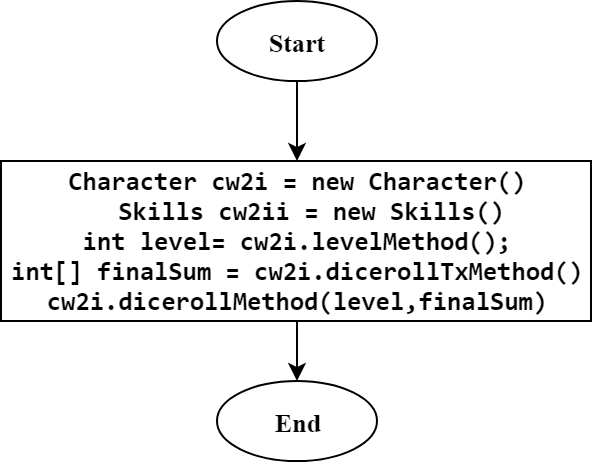
**Flowchart**



****

****

****

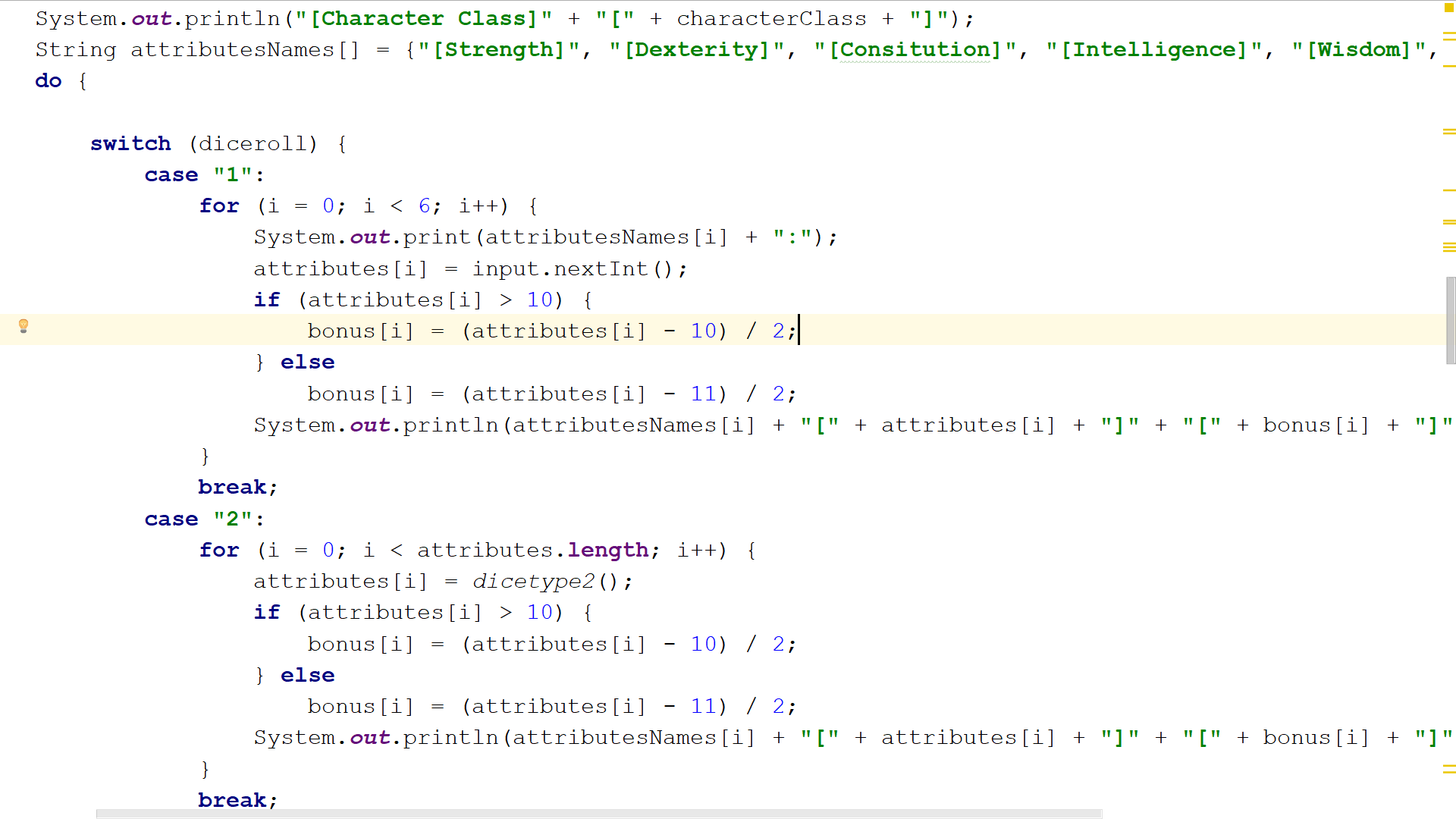


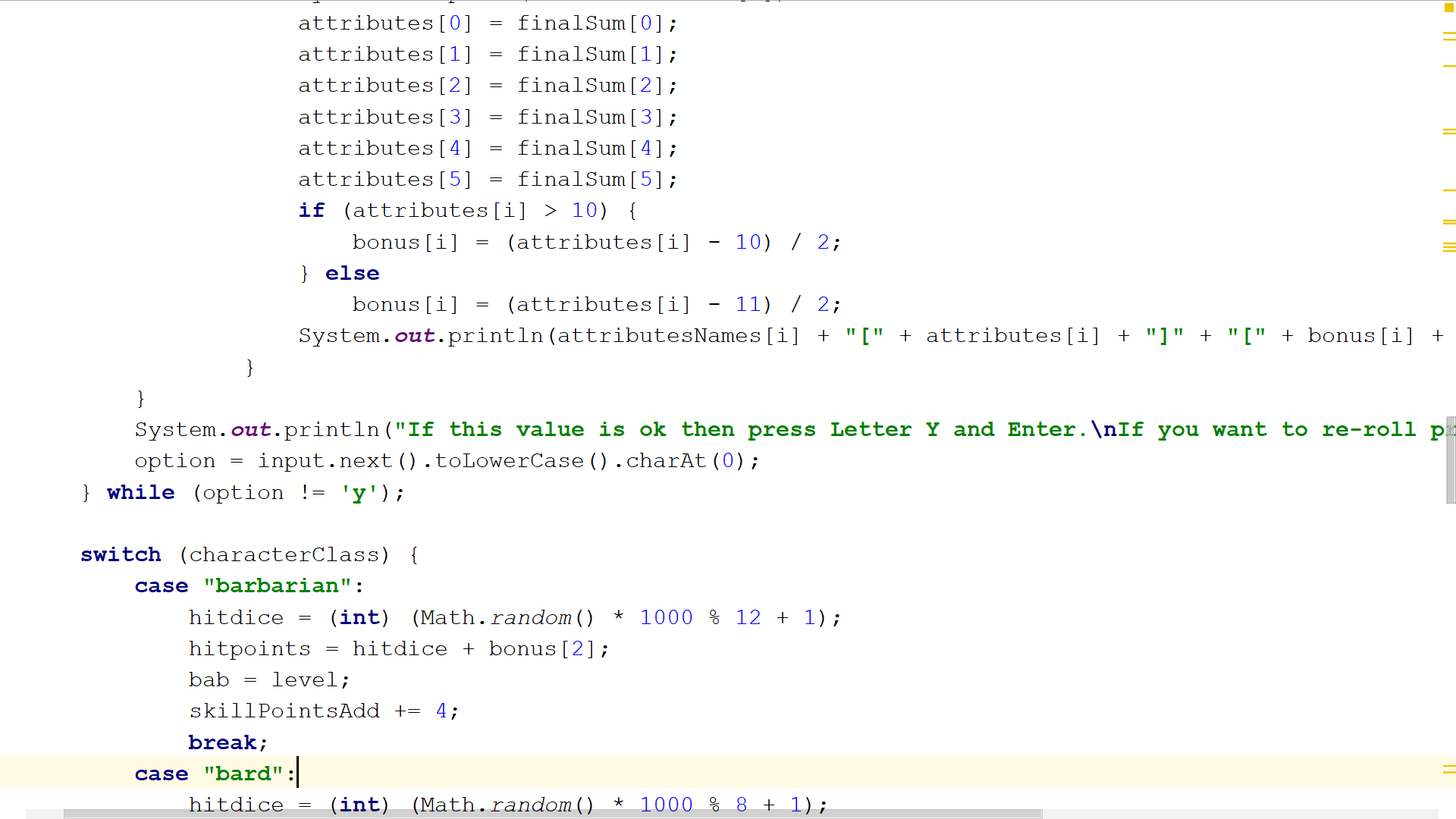
**Implementation**

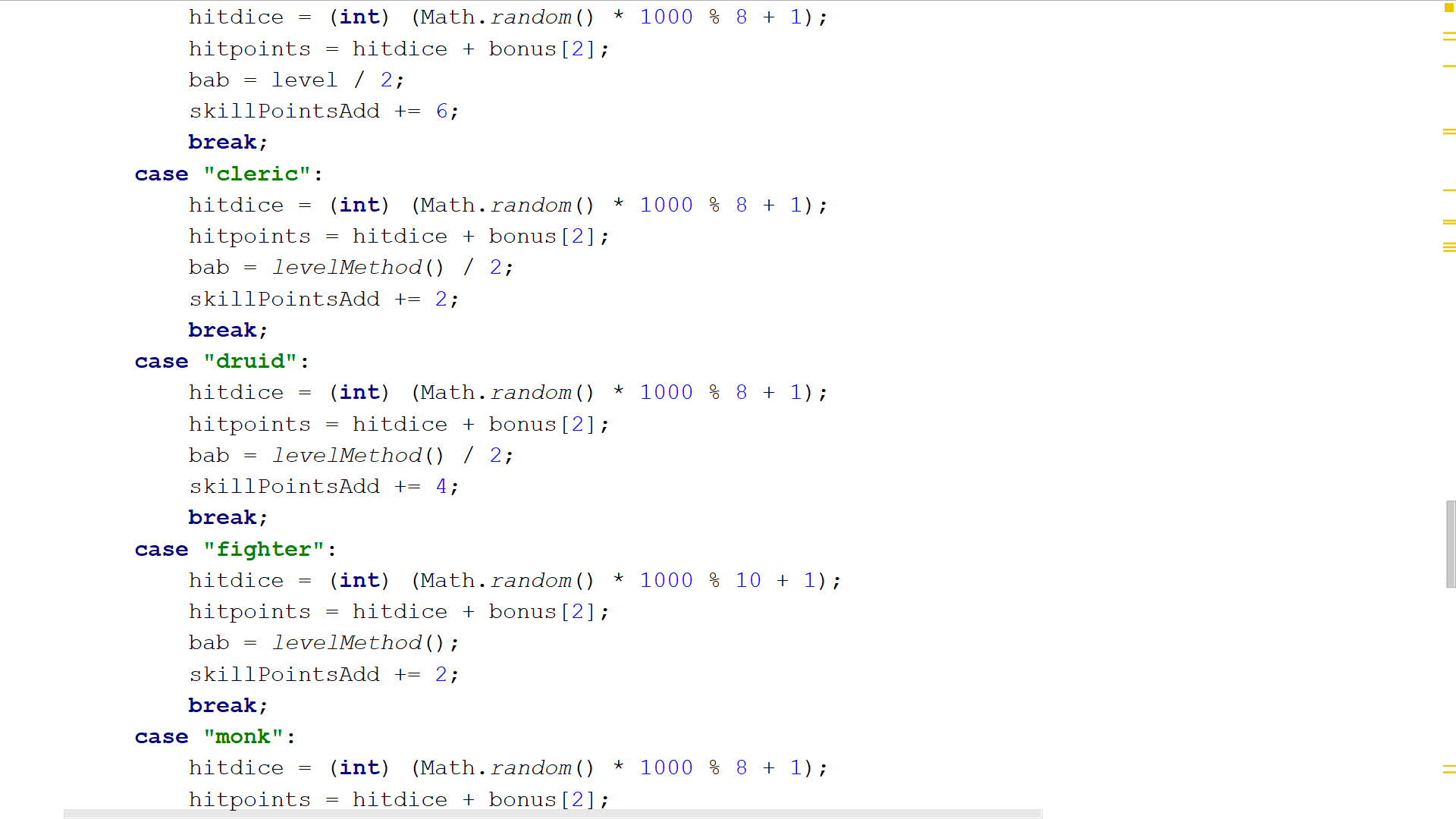


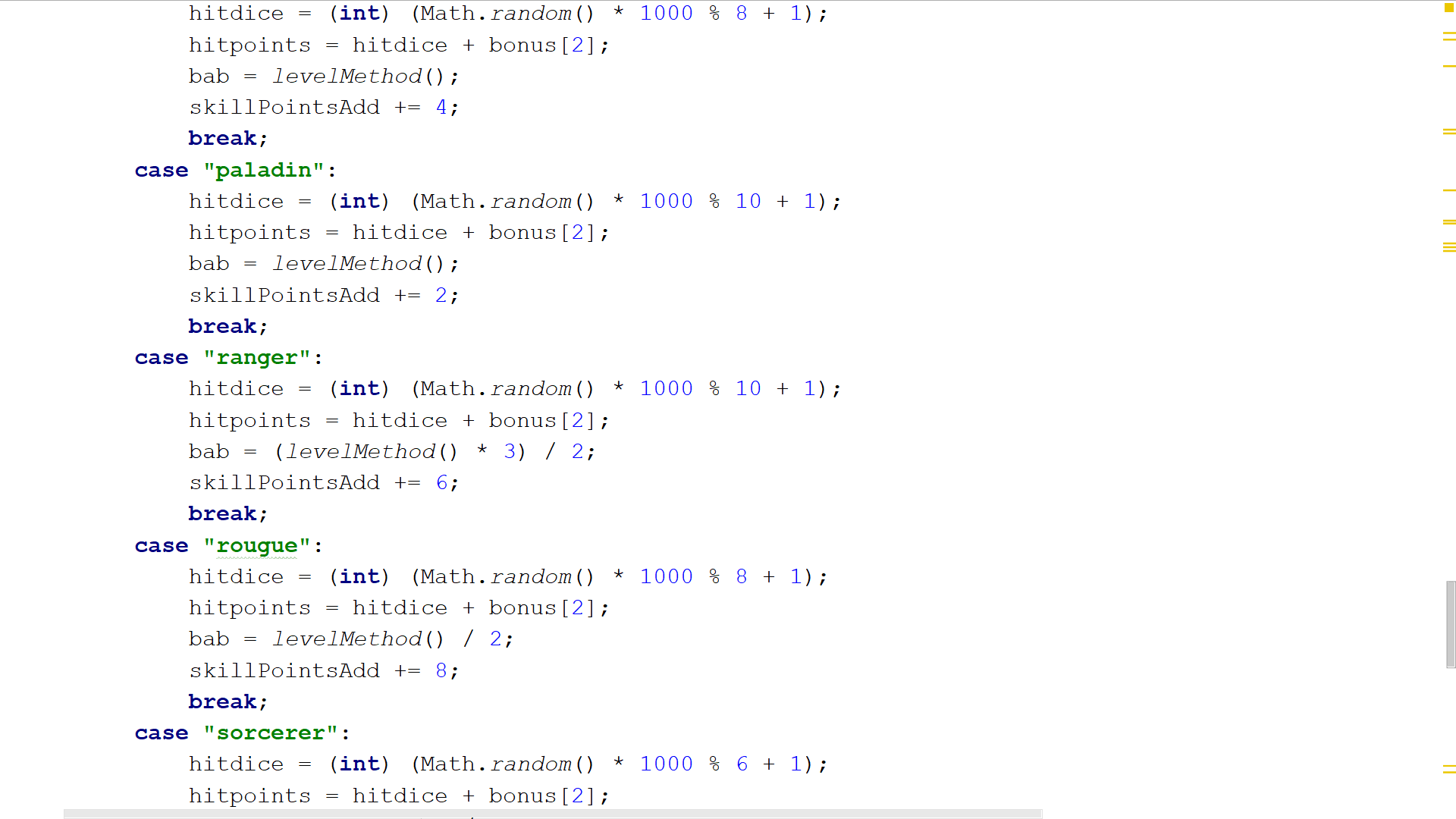


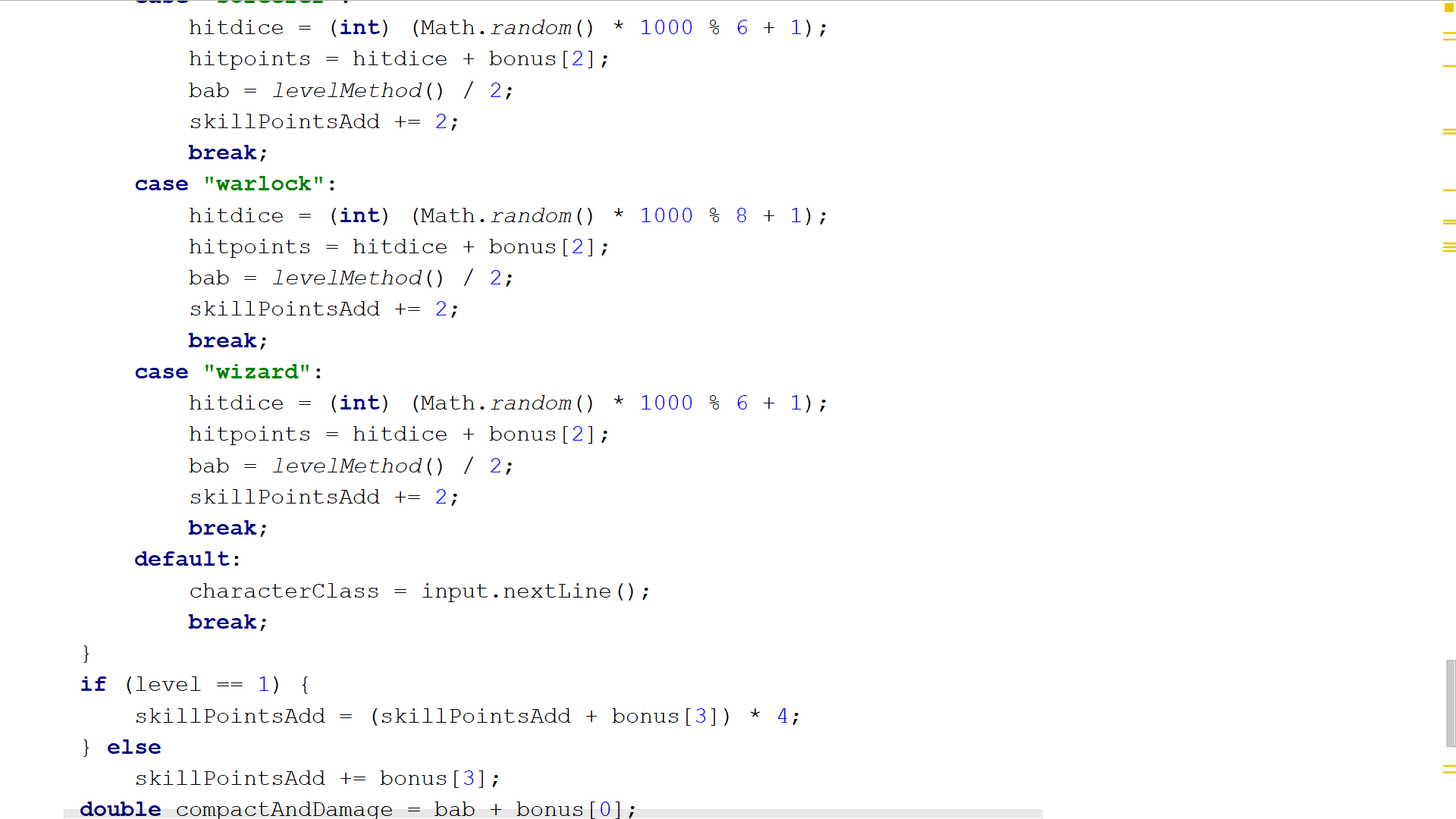




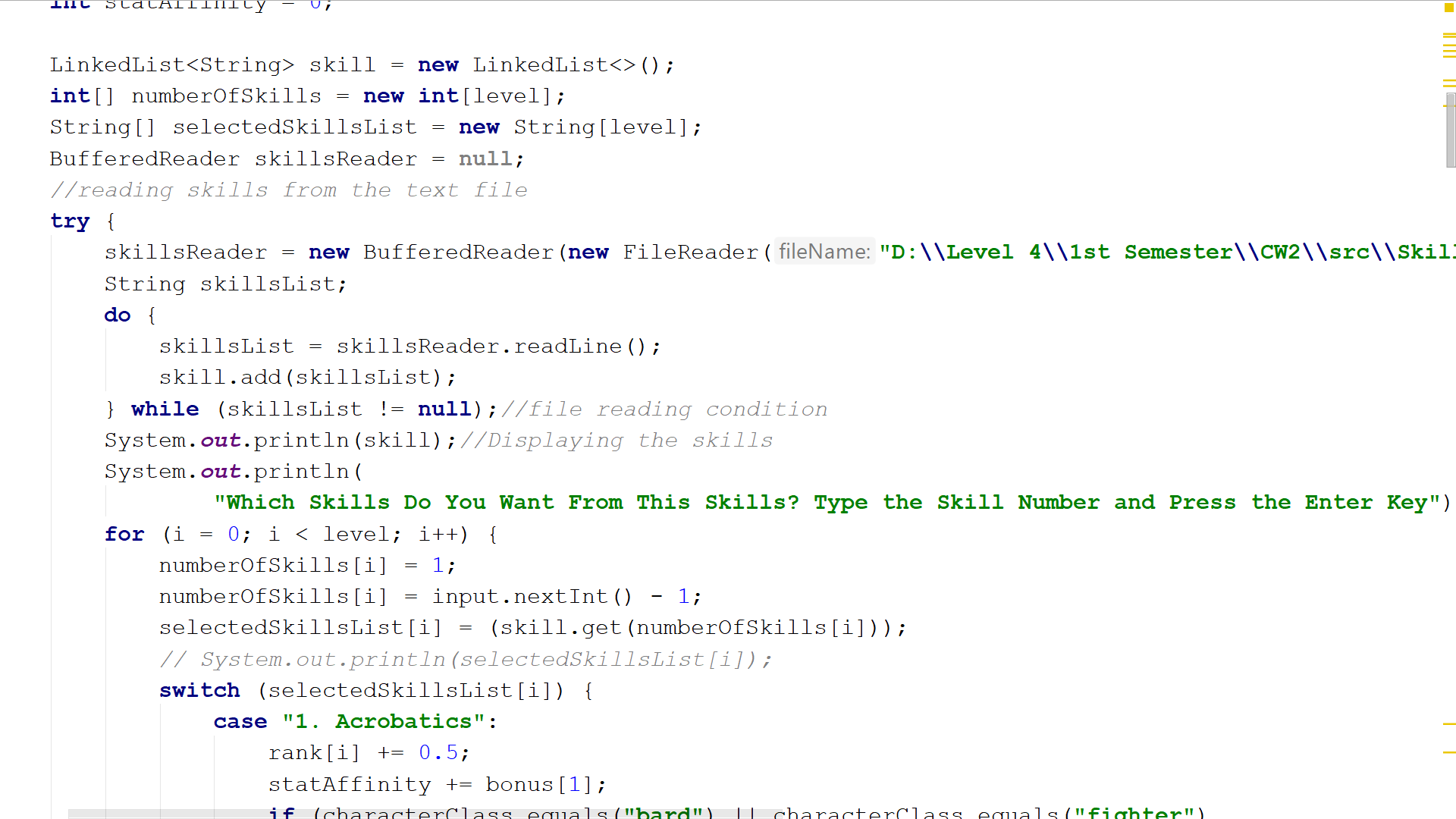




















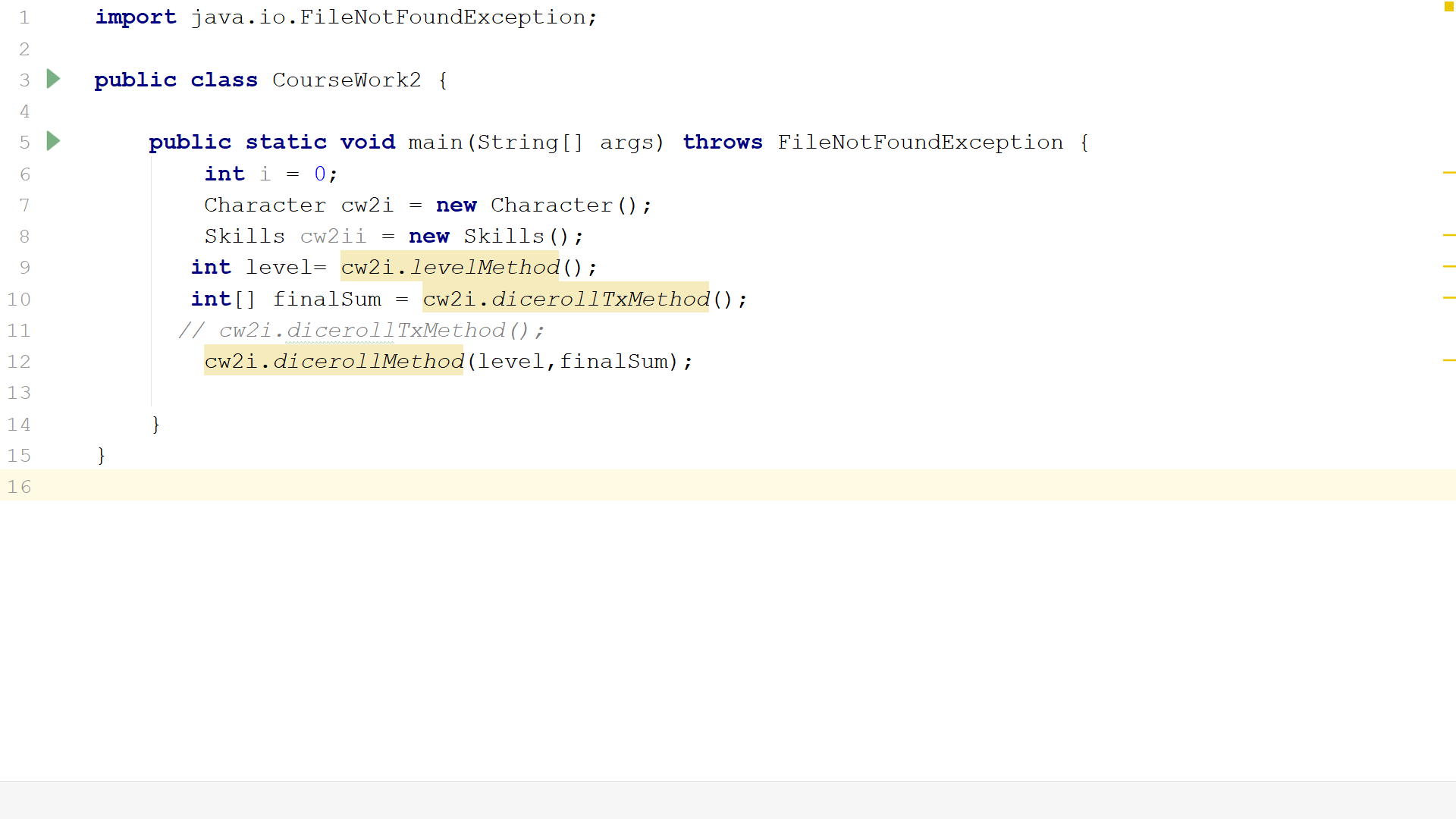






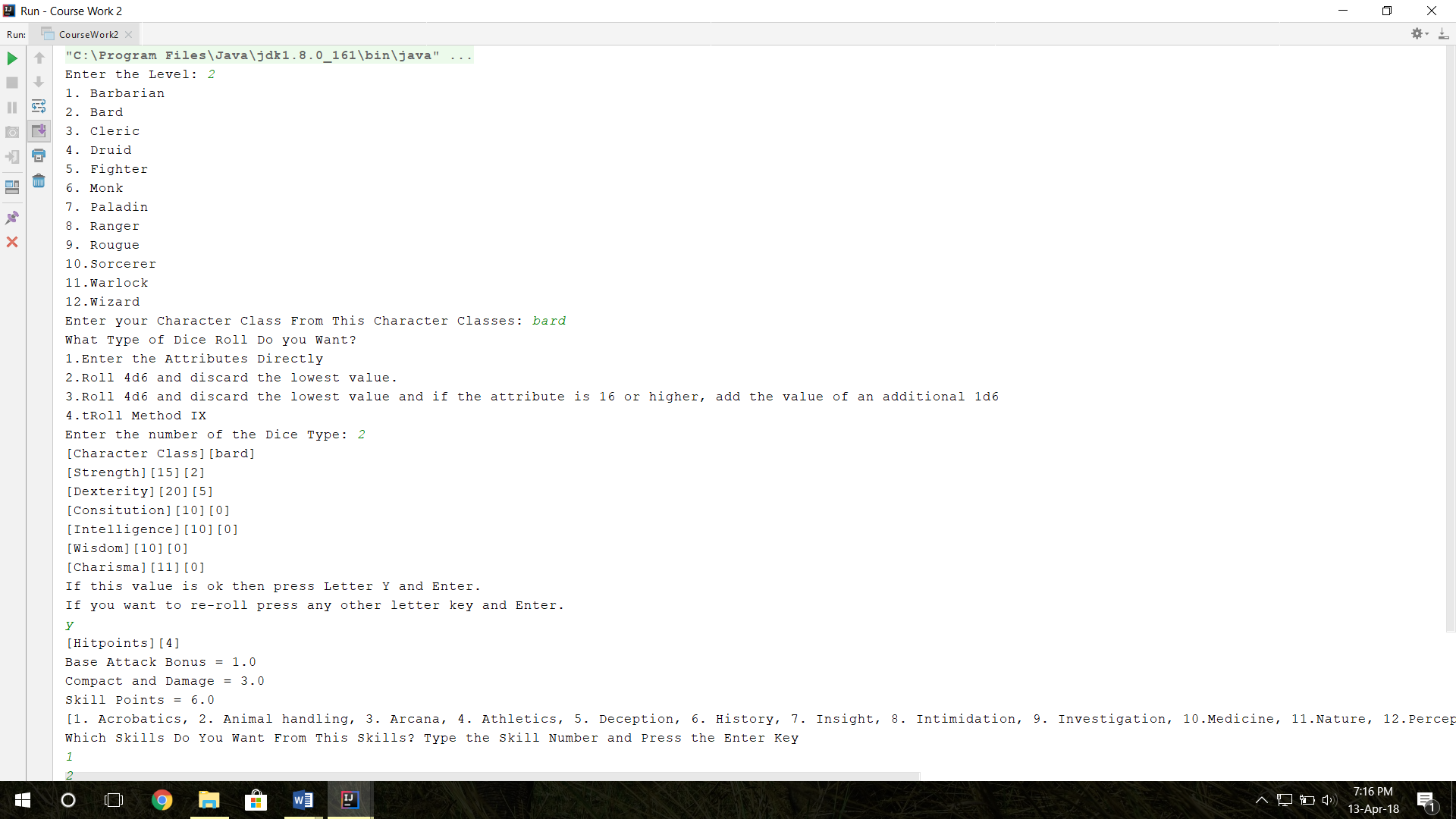


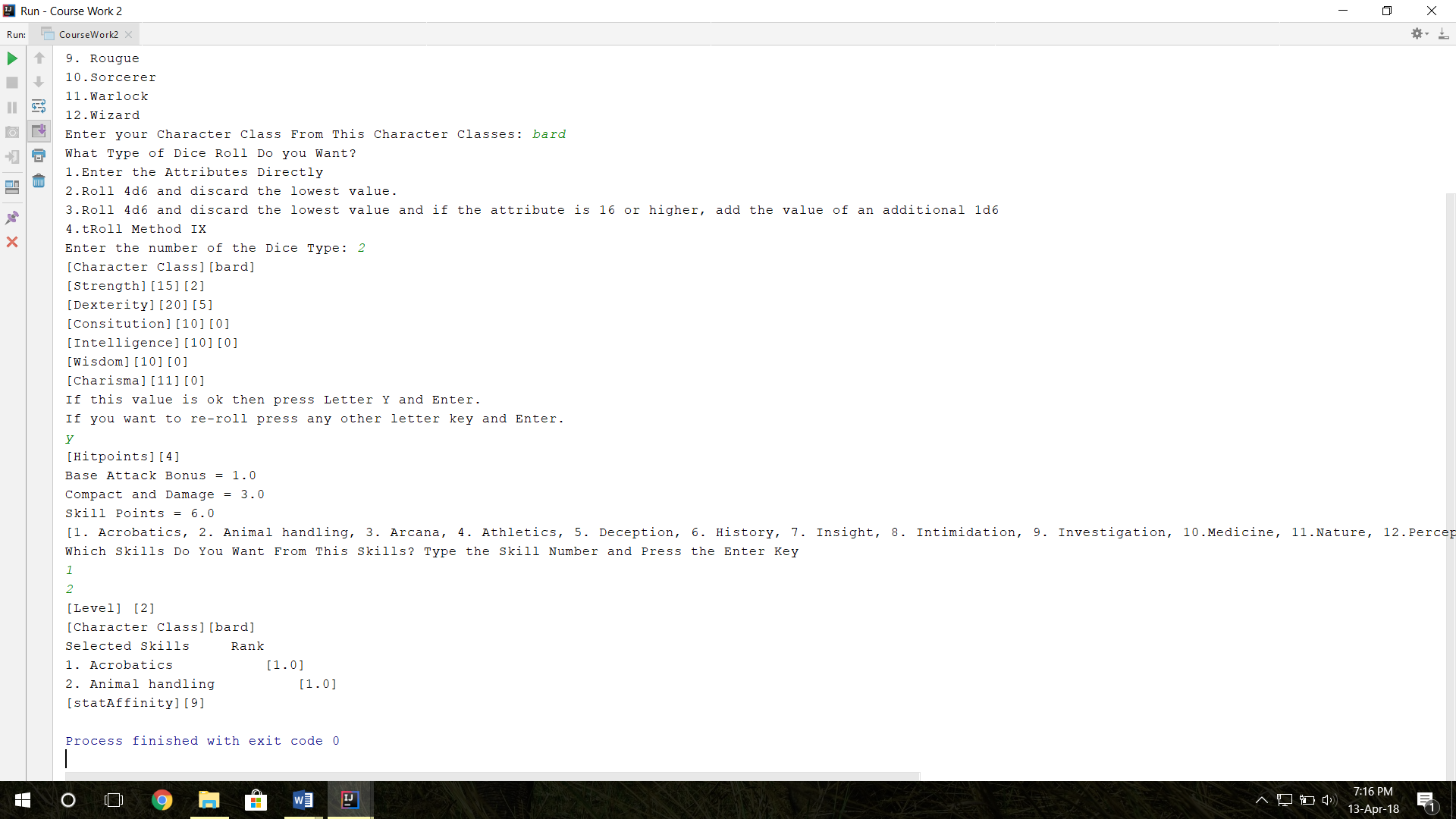




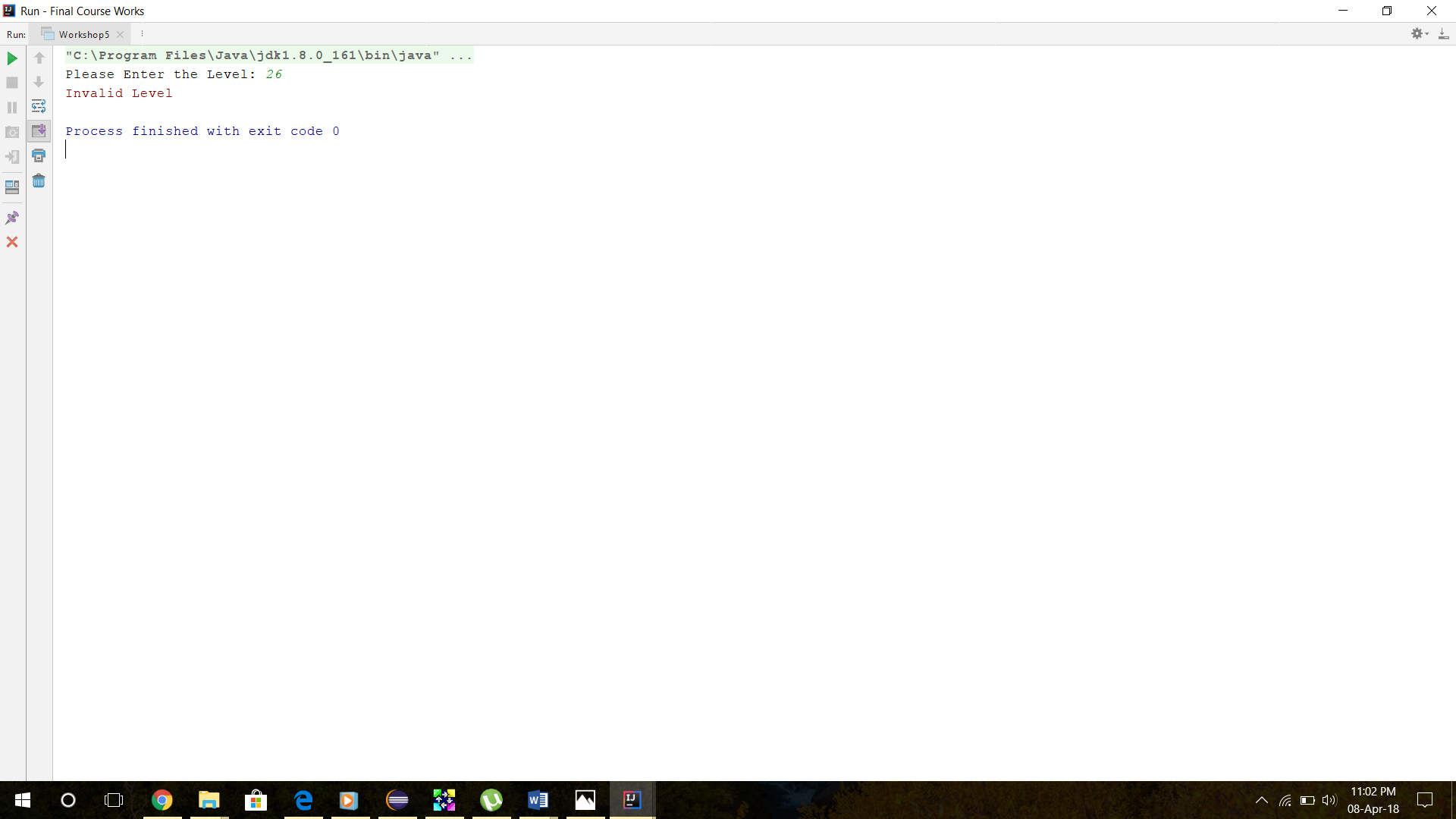
**Black Box Testing**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Data Set** | **Input** | **Expected Output** | **Output** | **Bug** |
| 1 | Level = 2  Character Class = Bard  Dicetype = 2  Skill 4  Skill 8 | Level = 2  Character Class = Bard  Athletics  Intimidation | Level = 2  Character Class = Bard  Athletics  Intimidation | No Bug |
| 2 | Level = 26 | Invalid Level  Game will exit | Invalid Level  Game will exit | No Bug |
| 3 | Level = 0 | Invalid Level  Game will exit | Invalid Level  Game will exit | No Bug |
| 4 | Level = -1 | Invalid Level  Game will exit | Invalid Level  Game will exit | No Bug |

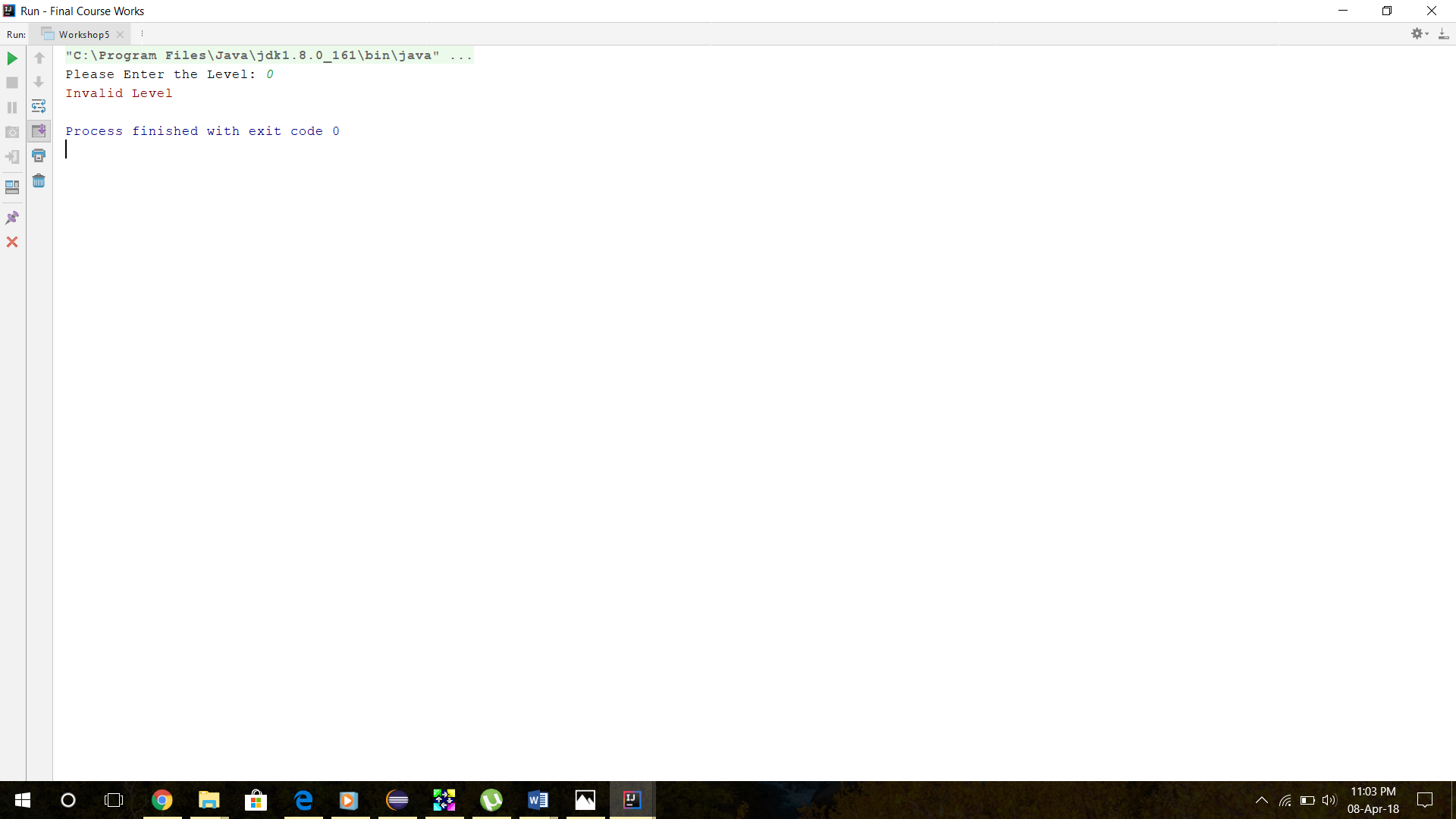
**Data Set 1** **Screenshot**



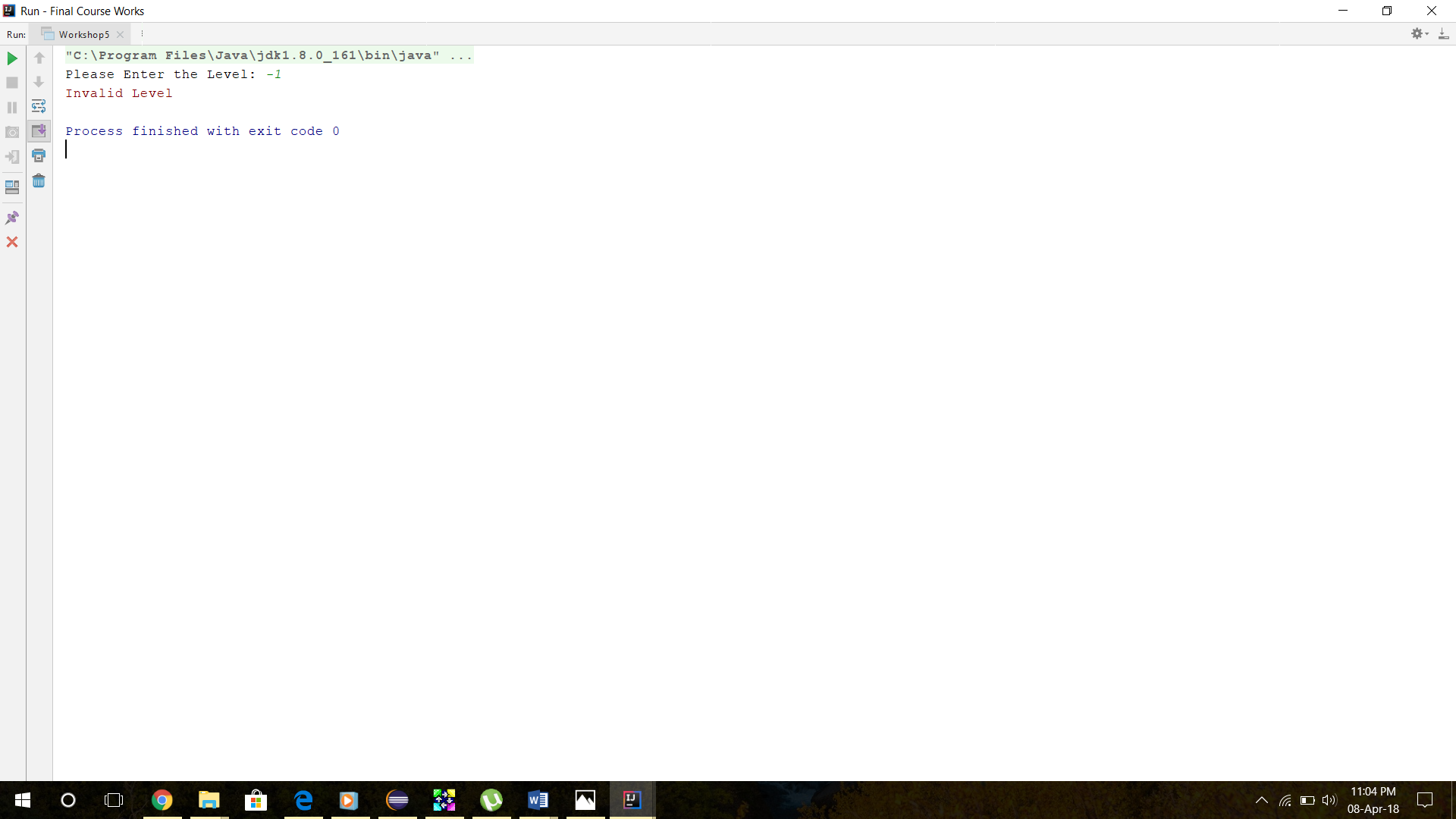
**Data Set 2 Screenshot**



**Data Set 3 Screenshot**



**Data Set 4 Screenshot**



**Conclusion**

According to this report I have concluded my ideas and designs the user can select dice type to roll the dice and get values. Then submit the values for the variables. The program go through the values and calculates the bonus for every single variable and finally the program gives the output and prints it.

When I prepared to do this I started the Course work 2 with no ideas of it so I googled most of the parts and I went through the players handbook and tested it. Finally I coded the program.

**References**

|  |  |
| --- | --- |
| [1] | Dungeons and Dragons player's handbook, Renton: Wizard of the Coast, 2014. |
| [2] | "Oracle," Oracle, [Online]. Available: <https://docs.oracle.com/javase/8/docs/>. |