



ALL THE BRAVE FRONTIER

GAME DESCRIPTION

Final Fantasy All the Bravest is a Final Fantasy game developed by BitGroove and published by Square Enix. Often regarded as the worst Final Fantasy game of all time, the game has received generally negative and scathing reviews, citing its blatantly greedy business model and almost non-existent gameplay. **Brave Frontier** on the other hand is a very similar game developed and published by gumi Asia Pte. Ltd. Developed in 2013, the game recently went viral, and had enjoyed a good following among its fans.

For this machine project, we are going to combine the best of both games to develop a new RPG style game. Uncreatively titled **All The Brave Frontier**, we will be taking the leveling and summoning system of Brave Frontier and combining it with the combat system of All the Bravest, adding, modifying and scoping down some of the features so that by the end of the term, you would have a simple but playable game that you can choose to continue to work on in the future.


CHARACTER CLASSES AND UNIT CREATION


There are 5 character classes in All the Brave Frontier, each with their own unique Main Abilities and Stat Distributions. There are, however, some common features among all the classes. All characters have a name. When a character is generated, a name selected at random from a list of names is assigned to that character. This list of names will be given to you later in a supplementary document.


All characters also have MAG, RES, ATK and CON, which decides their effectiveness in their roles. MAG is usually associated with magic abilities, the White and Black Mages are the classes that most effectively utilizes the MAG stat. RES is a character's resistance to magic abilities. ATK is associated with physical abilities and is best utilized by the Constable and the Rogue. CON is a character's resistance to physical abilities. CON also determines the starting HP of a character in a battle. The starting HP of a character is equivalent to the character's $CON * 250$.


All characters can Attack, which means they deal damage to the enemy character. The amount of damage is determined by the Attack type of the character. The Attack of Physical Attack type characters deal an amount of damage equal to the character's $ATK * 50$, while the Attack of Magical Attack type characters deal an amount of damage equal to the character's $MAG * 25$. Attacking does not have any cost, and is useful for whittling down the HP of the enemy character.


Every battle spawns for you a new character. When you program this, you may make this with as much fanfare as you want, as this is one of the exciting moments of playing this game. Whenever a unit is generated, the class of the character is then determined at random, along with the stats. Despite the randomness of the stats, however, the stats would still have to be in range of what is allowable for the class. These stat ranges, along with the Main Abilities and Attack Types of the classes are as follows:

WHITE MAGE			
	STAT	MIN	MAX
	MAG	2	8
	RES	2	5
	ATK	0	2
	CON	3	5
MAIN ABILITY: Divine Renewal – Heals all allies equal to this character's MAG + the ally's MAG * 50.			
EVOLUTION LINE:	Acolyte	Cleric	Holy Mage
ATTACK TYPE:	Magical		

BLACK MAGE			
	STAT	MIN	MAX
	MAG	4	10
	RES	3	8
	ATK	0	1
	CON	1	3
MAIN ABILITY: Astral Hurricane – Deals damage the enemy equal to the higher of 200 or this character's MAG * 100 – enemy's RES * 80, up to a maximum of 1000. This character's MAG is then reduced by 20%.			
EVOLUTION LINE:	Apprentice	Scholar	Wizard
ATTACK TYPE:	Magical		

RED MAGE			
	STAT	MIN	MAX
	MAG	4	6
	RES	3	8
	ATK	0	2
	CON	3	5
MAIN ABILITY: Spirit Bind – Increases the individual stats of all the party members by 10% of this character's individual stats respectively until the end of the battle. This cannot let the individual stat go beyond 100.			
EVOLUTION LINE:	Stage Hand	Assistant	Magician
ATTACK TYPE:	Magical		

CONSTABLE			
	STAT	MIN	MAX
	MAG	2	3
	RES	4	7
	ATK	5	8
	CON	8	12
MAIN ABILITY: Take and Receive – attacks the enemy and deals damage equal to the higher of 100 or this character's $ATK * 100 - \text{target's CON} * 20$ up to a maximum of 1000, then takes damage equal to the higher of 100 and the enemy's $ATK * 50 - \text{this character's CON} * 50$ up to a maximum of 500.			
EVOLUTION LINE:	Squire	Soldier	Knight
ATTACK TYPE:	Physical		


ROGUE			
	STAT	MIN	MAX
	MAG	1	4
	RES	1	8
	ATK	7	11
	CON	1	4
MAIN ABILITY: Blood-soaked Blades – attacks a target enemy and deals damage equal to this character's $ATK * 100$, and deals damage to itself equal to 25% of the damage dealt.			
EVOLUTION LINE:	Sewer Rat	Cutpurse	Cutthroat
ATTACK TYPE:	Physical		


ENEMY CHARACTERS


In this game, you will be battling against 4 kinds of enemies. The player selects which enemy to battle at the start of the game. Initially, the player has access only to the first two enemies; the player unlocks the next enemy after defeating both initially available enemies at least once, and unlocks the last enemy after beating the third enemy at least once as well.


All enemy characters have the same stats as the player, and the same actions, which means all enemy characters can either Attack or use their Main Ability (unless they have no Main Ability). Enemy characters also have an Attack Type, which determines the kind of damage the enemy character will deal. When attacking, enemies will deal their damage to ALL the player characters equal to the damage computation as dictated by their Attack Type.

When an enemy character is selected for battle, its stats are randomly generated, bounded by a minimum and a maximum value similar to how Player characters are created. The enemy's HP is computed by its $CON * 500$. Enemies also have Gambits, which is used in the Stacked Queue Turn Based Priority Combat System implemented in this game (see COMBAT SYSTEM for more details). The stats, abilities and Gambits of the enemies are as follows:

GOBLIN			
	STAT	MIN	MAX
	MAG	8	16
	RES	5	20
	ATK	2	8
	CON	12	20
MAIN ABILITY: Goblin Swipes – Deal damage to the target equal to the target’s ATK * 25 + this character’s ATK * 25			
ATTACK TYPE:		Physical	
GAMBITS:			
First in Queue		Attack	
Second in Queue		Attack	
Third in Queue		Skip	
Fourth in Queue		Main Ability – Goblin Swipes	
Fifth in Queue		Skip	

LIZARDMAN			
	STAT	MIN	MAX
	MAG	8	12
	RES	16	21
	ATK	20	21
	CON	21	23
MAIN ABILITY: Intelligent Strikes – Deal damage to its target equal to this character’s MAG * 100			
ATTACK TYPE:		Physical	
GAMBITS:			
First in Queue		Attack	
Second in Queue		Attack	
Third in Queue		Attack	
Fourth in Queue		Main Ability – Intelligent Strikes	
Fifth in Queue		Skip	

SKELETAL DRAGON			
	STAT	MIN	MAX
	MAG	23	27
	RES	20	27
	ATK	16	23
	CON	30	50
MAIN ABILITY: Boneshard Barrage – Deal damage to its target equal to ATK * 100			
ATTACK TYPE:		Magical	
GAMBITS:			
First in Queue		Attack	
Second in Queue		Attack	
Third in Queue		Main Ability – Boneshard Barrage	
Fourth in Queue		Main Ability – Boneshard Barrage	
Fifth in Queue		Skip	

DRAGONSIRE			
	STAT	MIN	MAX
	MAG	30	30
	RES	30	30
	ATK	30	30
	CON	70	70
MAIN ABILITY: Breath of Life – Deal damage to its target equal to this character’s ATK * 50 and heals this character equal to this character’s MAG * 100. Also increases this character’s individual stats by 10% respectively.			
ATTACK TYPE:		Magical	
GAMBITS:			
First in Queue		Attack	
Second in Queue		Attack	
Third in Queue		Attack	
Fourth in Queue		Attack	
Fifth in Queue		Main Ability – Breath of Life	

COMBAT SYSTEM

The combat of All the Brave Frontier follows a Stacked Queue Turn Based Priority System. The player and the enemy take alternating turns, with the Player starting first. One set of player and enemy turn is called a Round. The game is over if, at the start of a Round, either the enemy's HP has been brought down to 0, or all the HP of the player's unit have been brought down to 0.

During the player's turn, the player pushes all of his or her units one at a time into a stack, along with the action he or she would want the respective unit to do (i.e. Use Main Ability, or Attack). For example, if a party is composed of the following units:

				
MELCHIOR	BARTHOLOMEW	HIROKI	FARAH	LIEZLE
WHITE MAGE	CONSTABLE	NINJA	BLACK MAGE	BLACK MAGE

Note how the party does not have a Red Mage. The player can opt to not have a unit each for every class, however, the party must still have 5 units.

On the player's turn, the player can then choose to stack his or her units (along with their selected actions, the selection of the action is made while the units are being pushed into the stack) in this manner (assuming the Head of the stack is at the top):


HIROKI	Main Ability
MELCHIOR	Attack
FARAH	Main Ability
LIEZLE	Attack
BARTHOLOMEW	Main Ability
ACTION STACK	

After this is done, the units are popped one at a time and will be performing the action selected for it. After a unit is popped, it is then enqueued in a Damage Queue. The Damage Queue will determine the order by which the units of the player are attacked by the enemy character. Therefore, in our example, since Hiroki is the head of the stack, Hiroki will be popped first, and the unit will perform the action selected for it. Hiroki will be followed by Melchior performing an Attack, then Farah performing the Main Ability of the Black Mage, then Liezle with an Attack and finally Bartholomew with a Main Ability. They will then be Queued in the following manner in the Damage Queue (assuming the Head of the queue is at the top):

HIROKI
MELCHIOR
FARAH
LIEZLE
BARTHOLOMEW
DAMAGE QUEUE

After all the player's units have been popped from the stack and has been enqueued into the Damage Queue, the enemy then gets a turn. Following the Gambit listed under the enemy the player is currently fighting, the units in the Damage Queue are dequeued one at a time and either becomes the target of the enemy's attack or the enemy's Main Ability.

Assuming the team was fighting the enemy:

GOBLIN	
	
GAMBITS:	
First in Queue	Attack
Second in Queue	Attack
Third in Queue	Skip
Fourth in Queue	Main Ability – Goblin Swipes
Fifth in Queue	Skip


Since Hiroki is first in the Damage Queue, Hiroki will be attacked (First in Queue will be Attacked, according to the Gambits of the Goblin). Then Melchior will be attacked, followed by Farah being skipped, Liezle being hit by the Main Ability of the Goblin and finally, Bartholomew will be skipped.


If the Damage Queue does not have enough characters, then the rest of the enemy character's Gambits are skipped.

LEVELING SYSTEM


Leveling occurs by fusing units. You can only fuse two units if they belong in the same class. For example, you can only fuse a Wizard with an Apprentice but not a Trainee with a Sewer Rat. Each unit starts at Level 1 with the base stats of each class. Every time you fuse, the unit with the lower average stats gets fused to the unit with the higher average stats. In addition, the unit with the higher average stat levels up once, and increases its individual stats equal to 10% of the unit that was fused to it.

For example, given the following units:

ACOLYTE: Melissa (Level 2)		
	STAT	VALUE
	MAG	5
	RES	3
	ATK	1
	CON	2

ACOLYTE: Hannah (Level 1)		
	STAT	VALUE
	MAG	5
	RES	4
	ATK	1
	CON	3


The average of Melissa's stats is 2.75, while the average of Hannah's stats is 3.25; therefore Melissa fuses with Hannah. It doesn't matter that Melissa is of a higher level, since the average of Hannah's stats is higher than the average of Melissa's stats, then Melissa gets fused to Hannah. Hannah levels up and its stats increase to the following:


ACOLYTE: Hannah (Level 2)		
	STAT	VALUE
	MAG	5.5
	RES	4.3
	ATK	1.1
	CON	3.2

Whenever you need to display the stats of a character, show all the decimal places (up to 8).


When a first tier unit (Acolyte, Apprentice, Stage Hand, etc) reaches level 10, it evolves into a second tier unit. Whenever a unit evolves into a second tier unit, increase its individual stats by 20% of its individual stats.

For example:


ACOLYTE: Melchior (Level 9)		
	STAT	VALUE
	MAG	14
	RES	15
	ATK	5
	CON	17

ACOLYTE: Hannah (Level 2)		
	STAT	VALUE
	MAG	5.5
	RES	4.3
	ATK	1.1
	CON	3.2

If the two units are fused, Hannah gets fused to Melchior, because Hannah has less average stats than Melchior. This makes Melchior level 10, with the following stats:

ACOLYTE: Melchior (Level 10)		
	STAT	VALUE
	MAG	14.55
	RES	15.43
	ATK	5
	CON	17

Since Melchior is now level 10, it will also evolve into a level 10 Cleric with the following stats:

CLERIC: Melchior (Level 10)				
	STAT	OLD VALUE	20%	NEW VALUE
	MAG	14.55	2.91	17.46
	RES	15.43	3.086	18.516
	ATK	5	1.022	6.132
	CON	17	3.464	20.784

Similarly, when a second tier unit (Cleric, Scholar, Assistant, etc.) reaches level 30, it evolves into a third tier unit and increases its individual stats by 30% of its individual stats. The maximum level for any character is 50. See the **CHARACTER CLASSES** section to see how the Base Stats of every character is determined.

FULL GAME MECHANICS

1. Start the game by asking for the player's name. This should be used every time you wish to address the player.
2. Then, randomly generate the 25 characters of the player, following the rules set in the CHARACTER CLASSES AND UNIT CREATION section. Do not make this process take too much time (especially in the way you display the generated units), as it might make the demo take a long time.
3. Once the twenty five characters of the player have been created, have the player select five of the twenty five units to include into his or her party.
4. Then, have the player select one of the two starting enemies to fight.
5. After the match, take the player back to the main menu. The contents of the main menu would vary, depending on the following factors:
 - a. If the player has defeated both of the starting enemies at least once, an option to battle Skeletal Dragon should be present.
 - b. If the player has defeated the Skeletal Dragon, an option to battle the Dragonsire should be present.
 - c. If the player has more than 5 characters, the option to fuse units should be present. Remember that a player can only have 5 characters in their party, and cannot have less than 5.
 - d. An option to end the game should also be present at all times.
6. The player must be able to continue battling and unlocking characters and enemies until the player decides to quit.
7. To facilitate the MP demo, when developing your MP, proper follow modularization so that modifications can be easily made.

MP DELIVERABLES AND MILESTONES

For the first milestone of your MP, you are required to create the working Battle System implementation, where all the features of the Battle System are present and functional. Since no Leveling System is present yet, it is not expected that the player characters will be able to beat the Skeletal Dragon or the Dragonsire. For the sake of testing, however, enable the combat for these two enemies at the start of the game (as opposed to only enabling the Goblin and the Lizardman).

For the UI, a simple text based UI that will facilitate all the Leveling System requirements will do.

For the second and final milestone of your MP, you are required to finish all the requirements of this MP. A text based UI may be created to facilitate the whole MP, but if you can research on how to create a GUI, then that would be good as well. Whichever the case may be, the UI must be functional and useable, and all the requirements must be met and be visible on the UI created.

GENERAL REQUIREMENTS

The following general set of guidelines should be followed:

1. The implementation of this project will require you to use Java to be compiled and ran with Eclipse, or the command prompt compiler.
2. You are required to create and use methods and classes whenever possible. Make sure to modularization principles properly. No brute force solution.
3. The above description of the program is the basic requirement. Any additional feature will be left to the creativity of the student. Bonus points would be awarded depending on the additional implemented features. These additional features could include new character classes, new enemies, player characters also having Gambits, etc. Depending on the scale of the new feature, additional points will be awarded to the student. However, make sure that all the minimum requirements are met first; if this is not the case then no additional points will be credited despite the additional features.
4. For the minimum requirements of this MP, Battles, Unit Generation and Unit Leveling (via fusing), following the rules and requirements mentioned in the relevant chapters, should be present and working. A text based interface would do.
5. Do not forget to include internal documentation (comments) in your code. At the very least, there should be an introductory comment before every method.
6. Statements and methods not taught in class can be used in the implementation. However, these are left for the student to learn on his or her own.
7. Submission of the project is on the indicated submission deadlines. Late submissions **WILL NOT BE ACCEPTED**. A signed copy of the proof of submission will be returned to you.
8. Any instruction not followed will incur deductions (or a 0.0).
9. This project is individual. Copying other people's work and/or working in collaboration (i.e. sharing code) with other students are **not** allowed and are punishable by a grade of **0.0** for the entire DASTRAP course and a case may be filed with the Discipline Office. In short, do not risk it; the consequences are not worth the reward. (Also, what is a measly passing grade compared to a life-long conscience burden?)

10. During the MP demo, the student is expected to appear on time, to answer questions in relation to the output and to the implementation (source code) of the project, and/or to revise the program based on a given demo problem. Failure to meet either of these requirements could result to a grade of **0.0** for the project.
11. It should be noted that during the MP demo, it is expected that the program can be compiled successfully and will run. If the program does not run, the grade for that phase is 0. However, a running program with complete features may not necessarily get full credit, as implementation (i.e., code) will still be checked.

SUBMISSION CHECKLIST FOR EACH PHASE

- CD (placed in a case) or flash drive, either of which should be properly labeled with name and section, containing the following files:
 - The Java project
 - The function specs document
 - any additional libraries you used for your implementation

Take note that in case of a flash drive submission, the flash drive **may not** be returned.

- Printout of the following:
 - The Java source files
 - The UML Class Diagram
- Short brown envelope containing all the above requirements, with 2 copies of the proof of submission. One of the proof of submission is attached at the upper right corner of the back of the envelope (not the side with the flap). Each should indicate the following information:

Name	:
Section	:
Submitted To	:
Received by	:
Date / Time Received	:
- Email the Java project as an attachment to YOUR own email address and to the email address of the instructor (thomasjamestiamlee@gmail.com) on or before the deadline (to serve as back-up). Take note that the actual submitted work would take precedence over the email back-up, and in the case of discrepancies, the actual submission would be considered.

PROOF OF SUBMISSION

The proof of submission should be printed on a one fourth sheet of short bond paper, and should contain the following:

Name :
Section :
Submitted To :
Received by :
Date/Time Rcvd. :

As specified, one copy is to be attached in the upper right hand corner of the back of the envelope you used for your submission, and another copy would be used as your proof. Do not lose your copy, as this would act as your ticket to the MP demo. The Date/Time Rcvd. field can be filled in by hand.