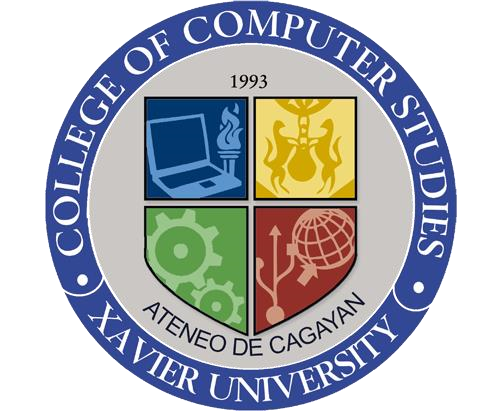
**BUNO**

presented to the

College of Computer Studies

Xavier University



Maria Liza O. Paano

Carl Louis D. Fabre

Kent Letrell A. Bigno

BUNO

Maria Liza O. Paano

BSCS-1   
Mangagoy Bislig City, Surigao del Sur  
Burgos Street, CDO  
+639278771802

lizapaano@gmail.com

Carl Louis D. Fabre

BSIT-1  
406 E Spencer, Kansas, USA  
12-20th Street, Nazareth, CDO  
+639194530905

Carlfabre1996@gmail.com

Kent Letrell A. Bigno

CompStud Department  
Arroville Subd Blck 16 Lt 23, CDOC  
Bolonsiri, Camaman-an, CDOC  
Telephone number, incl. country code

kentletrell@gmail.com

# TABLE OF CONTENTS

1 Introduction

1.1Overview

1.2Objectives

1.3Scope and Limitation

1.4 Functionalities

2. Program and Design Implementation

2.2 Pseudocode

2.3 Data Structure and Algorithms Discussion With Code Snippets

3. Conclusion

4. References

5. Appendices

5.1 Project Proposal

5.2 Photo Documentation/ Program Screenshots

# INTRODUCTION

Buno is a card game project by our group that was inspired by the Filipino card game “Bulak Bulak” which consist of 2 players.

## OVERVIEW

Buno is a card game that contains two players. Each player has 5 cards to begin the game. With a total of thirty-six cards, the first player to have zero cards wins the game. The deck contains cards with four colors from red, blue, white and yellow with each color with a numbering system from one to nine with a total of 36 cards. For the other player to not see the cards, the cards will be hidden when it’s not their turn.

## OBJECTIVES

The game objective is to withdraw all the cards in the player’s possession before the other player. Come up victorious against your opponent by strategizing on which colors with a high card value to drop. Valid cards will be played if it has the same color from the previous card. This game will be played by rounds. If you win the round, you keep your turn and can play a new color that you think your opponent lacks. To keep your turn and win rounds, your card value should be higher than your opponents. You have the freedom to choose if you wish to stick to that color or put in a new one, whichever you think your opponent lacks after winning rounds. Strategize your way to victory.

## SCOPE AND LIMITATION

The group will take the buno card game as a model for the project and imply the features needed.

We will run the card game through the command prompt as it’s the only available knowledge we have through teaching in class.

limitations

If drawn too many cards, the show board will not be properly due to the card decorations will not align.

The game also finishes fast as it only has a few amount of cards.

Since the game is played on command prompt and on the same screen, even with the hide card method, the cards can still be seen through turns. So in order to play this the correct way each player must not look if it’s their opponent’s turn

## FUNCTIONALITIES

For the functionalities of the game, there will be 36 cards in the deck. Five cards will be given to each player randomly before the game starts. A single card will be placed for the players to recognize what valid card to choose to start the game. The goal is to have zero cards left to win the game. Each player will put a card base on the color from the card pile for the move to be valid. Once both players have played a card, if the card value is greater than the other, that will determine whose turn it is to play. Therefore, this game does not have a vice versa type of function unlike most games.

# PROGRAM AND DESIGN IMPLEMENTATION

## PSEUDOCODE

Playgame(Player P, Player other){

If player has no same color on current card, card pick = null;

While(!hascolor(p)){

Player pick card from deck;

}

When player picks a valid card, showcard();

}

While pick is not valid{ prints “invalid pick, please pick a valid card”

}

card value {

if current value is > play value set token to false which indicated the player will lose the round

else if current value < play value set token to true which indicates, player wins round and still has turn.

}

## DATA STRUCTURES AND ALGORITHMS DISCUSSION WITH CODE SNIPPETS

The Data Structure involved in this program is an Array that stores the cards in the deck and Array list for the player cards and card pile. We used an array for the deck because we know the total number of cards, which is thirty-six cards. We used an array list for the player cards and card pile because the number of cards on both are not fixed and is dependent on the circumstances.

Algorithm Discussions

The game has five classes from cards, deck, player, buno and main. We separated each of these functions to organize and make it to locate the functions that was put in each class. Each class has their own attributes that can be called later in the buno class which is where the main game occurs. The deck then has arrays for the value and color. We used an array on this class to set a fix amount on the total number of cards. Then we used a for loop to put all elements onto the deck. The deck also contains methods to check if it’s empty, card peek, get top card and add to deck. On the players class, we used again an array list because the number of cards is not a fix number. It also has methods to show and hide the cards between turns and check if a player has won the game or not.

The buno class is where all the classes and their implementations takes effect. This is where the playGame method is implemented. The player has to have the same color as the one on the current card in order for that pick to be valid. If not, the player will pick from the deck until a valid card is received. The player turns is not like most card game where you take turns vice versa. Our turn system is based on the card value. If the card value is greater than the current card, then the player wins the round and keeps their turn and can bypass the same color rule. Once the player wins the round, they can then put any color they want. The main method then calls the class buno.

Because our turns are not vice versa, this was the code used

Card play = p.throwCard(pick); //compare value

if (current.value > play.value) {

p.setToken(false);

other.setToken(true);

}

else if (current.value < play.value) {

p.setToken(true);

other.setToken(false);

}

If current value is greater than play value then token is set to false, else if current value is less than play value set token to true.

public void game() {

playGame(p1, p2);

playGame(p2, p1);

while(!gameOver(p1,p2)) {

if(p1.hasToken(p1)) {

setCurrent(p1);

playGame(p2, p1);

}

else if (p2.hasToken(p2)) {

setCurrent(p2);

playGame(p1, p2);

}

}

}

So, if token is true set current enables the player to play any color of their choice and keeps their turn.

# CONCLUSION

The game has its flaws. Although it’s fast paced it’s still fun to play and it requires trust on the two players because the cards can still be seen even with the hide cards method because it’s still played on the same screen. This game will eliminate most of the flaws if it’s played on two different devices by both players, but that’s still far from our knowledge. This buno game project was inspired by the filipino game bulak bulak.

# REFERENCES

https://www.baeldung.com/java-arraylist

https://stackoverflow.com/questions/20926835/card-game-in-java-creating-a-hand-of-cards

https://github.com/tribikram-adhikari/Simple-Text-Based-Uno-game-in-JAVA

https://www.callicoder.com/java-arraylist/

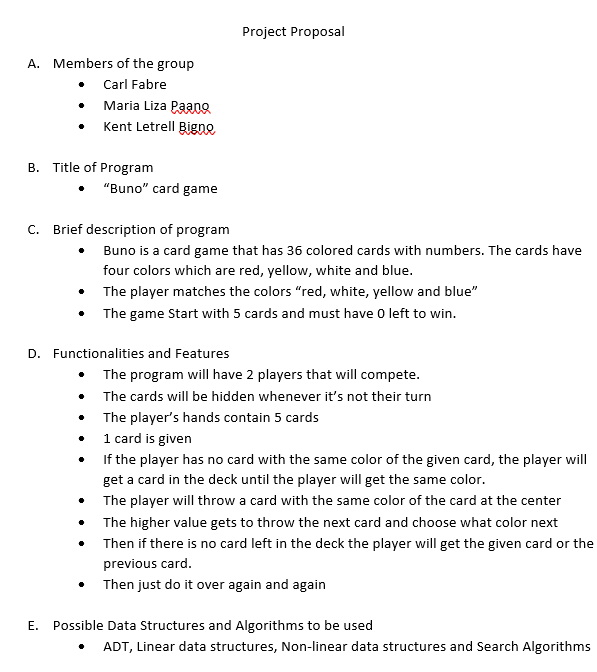
https://www.tutorialspoint.com/java/java\_arraylist\_class.htm

https://stackoverflow.com/questions/22133845/java-card-game-creating-the-cards

https://stackoverflow.com/questions/35764328/java-card-game-logic

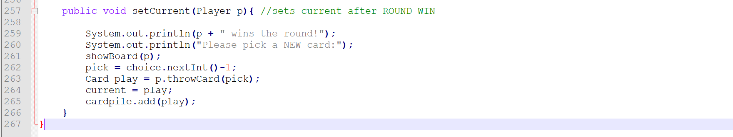
# APPENDEICES

## PROJECT PROPOSAL

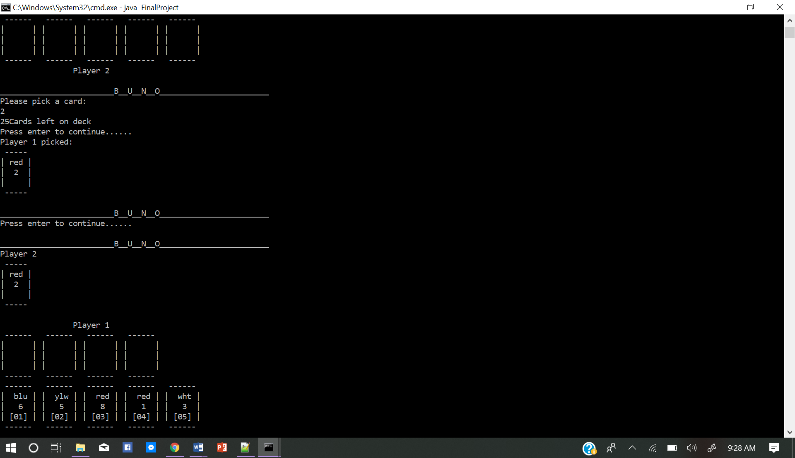
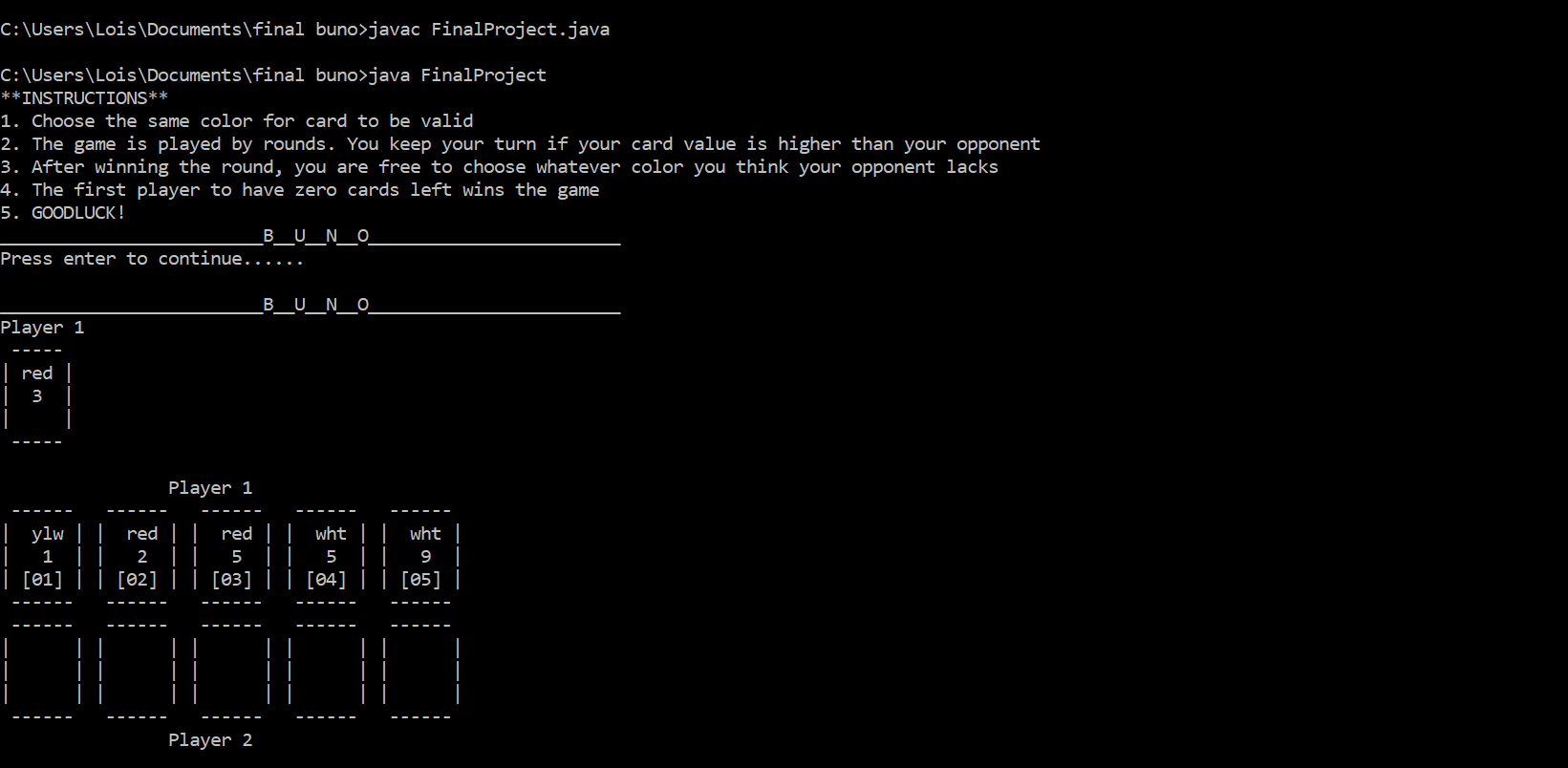


## PHOTO DOCUMENTATION/PROGRAM SCREENSHOTS

Players can bypass the “same color” rule and can put a new color after winning rounds



# This is the showboard, what players will see most of the time while playing the game

The game in cmd