Lab07 – Methods and Top-down Design

Enrique Saracho Felix
100406980
CPSC 1150
01/07/2023

Part A

Program WarGame

File name: WarGame.java

Purpose: To simulate a game of cards where the user goes against the computer.

Packages: java.util.Scanner

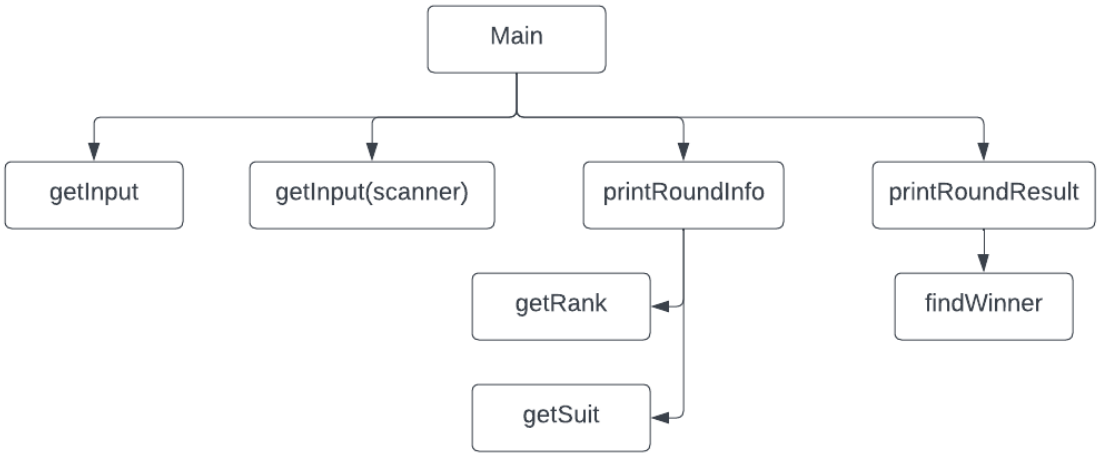
Limitations: The program will create an error message and prompt the user to enter again if the input is not an integer between 1 and 52 (inclusive).

Bugs: The compiler confuses some of the characters in the printRoundResult method.

Input: An integer from 1 to 52 for each round (there's 10 rounds in total).

Output: Several messages displaying the relevant information about the game and the results.

Design diagram:



Pseudocode:

Algorithm WarGame
START

(**main**)
Set *ROUND* = 10
Set integer *user*
Set integer *comp*
Set integer *score* = 0
Set integer *r* = 1
While (*r* < *ROUND*) {
	comp = *genInput*()
	user = *getInput*()
	printRoundInfo(*r*, *user*, *comp*)
	score = *printRoundResult*(*score*, *user*, *comp*)
}
Print “End of Game!”

(**getInput**)
Set integer *user*
Set boolean *flag*
Do {
	If (*flag*) {
		Print error message
	}
	Print “Enter card”
	Read *user*
	flag = true
} While (*user* < 1 || *user* > 52)
Return *user*

(**getInput**)
Set integer *comp*
comp = Math.random * 52 + 1
return *comp*

(**printRoundInfo**, parameters: *r*, *user*, *comp*)
Print “R” + *r* + “: “
Print “Computer card is “ + *getRank*(*comp*) + “ of “ *getSuit*(*comp*)
Print “; User card is “ + *getRank*(*user*) + “ of “ *getSuit*(*user*)

(**printRoundResult**, parameters: *score*, *user*, *comp*)
Set integer *winner* = *findWinner*(*user*, *comp*)
If (*winner* = 0) {
	score *= 2
	Print “It’s a tie…” + *score*
} else if (*winner* = 1) {
	score += 1
	Print “User wins…” + *score*
} else {
	Print “Computer wins…” + *score*

```
}
Return score

(findWinner, parameters: user, comp)
Set integer userCard
Set integer compCard
userCard = (user – 1) % 13 + 1
compCard = (comp - 1) % 13 + 1
If ( userCard = compCard ) {
        Return 0
} else if ( userCard > compCard ) {
        Return 1
} else {
        Return -1
}

(getRank, parameter: card)
Set string rank
Set integer num = (card – 1) % 13 + 1
If ( num = 1 ) {
        rank = “Ace”
} else if ( num = 10 ) {
        rank = “10”
} else if ( num = 11 ) {
        rank = “Jack”
} else if (  num = 12 ) {
        rank = “Queen”
} else if (  num = 13 ) {
        rank = “King”
} else {
        rank = (string) num
}
Return rank

(gerSuit, parameter: card)
If ( card < 14 ) {
        Return “Hearts”
} else if ( card < 27 ) {
        Return “Diamonds”
} else if ( card < 40 ) {
        Return “Clubs”
} else {
        Return “Spades”
}

END WarGame
```

Test run(s):

```
$ java WarGame.java
==> Enter your card: 0
Error: invalid input. Try again
==> Enter your card: 53
Error: invalid input. Try again
==> Enter your card: 10
R1: Computer card is King of Spades; User card is 10 of Hearts
Computer wins ? user score is 0
==> Enter your card: 21
R2: Computer card is 7 of Diamonds; User card is 8 of Diamonds
User wins ? user score is 1
==> Enter your card: 5
R3: Computer card is 8 of Diamonds; User card is 5 of Hearts
Computer wins ? user score is 1
==> Enter your card: 36
R4: Computer card is 7 of Diamonds; User card is 10 of Clubs
User wins ? user score is 2
==> Enter your card: 42
R5: Computer card is 6 of Spades; User card is 3 of Spades
Computer wins ? user score is 2
```

```
==> Enter your card: 50
R6: Computer card is Ace of Diamonds; User card is Jack of Spades
User wins ? user score is 3
==> Enter your card: 37
R7: Computer card is Jack of Hearts; User card is Jack of Clubs
It?s tie ? user score is 6
==> Enter your card: 39
R8: Computer card is 9 of Spades; User card is King of Clubs
User wins ? user score is 7
==> Enter your card: 25
R9: Computer card is King of Spades; User card is Queen of Diamonds
Computer wins ? user score is 7
==> Enter your card: 16
R10: Computer card is 3 of Clubs; User card is 3 of Diamonds
It?s tie ? user score is 14
End of Game!
```

Part B:

1. Changed line 22 from:

    comp = *genInput*()

to

    comp = *WarGame.genInput*()

Results:

- The program compiled and ran successfully.

```
$ java WarGame.java
==> Enter your card: 52
R1: Computer card is 5 of Clubs; User card is King of Spades
==> Enter your card: 1
R2: Computer card is 10 of Hearts; User card is Ace of Hearts
Computer wins ? user score is 1
==> Enter your card: 25
R3: Computer card is 9 of Diamonds; User card is Queen of Diamonds
User wins ? user score is 2
==> Enter your card: 34
R4: Computer card is 2 of Spades; User card is 8 of Clubs
User wins ? user score is 3
==> Enter your card: 26
R5: Computer card is 2 of Hearts; User card is King of Diamonds
User wins ? user score is 4
==> Enter your card: 29
R6: Computer card is 10 of Clubs; User card is 3 of Clubs
Computer wins ? user score is 4
==> Enter your card: 17
R7: Computer card is 2 of Spades; User card is 4 of Diamonds
User wins ? user score is 5
==> Enter your card: 14
R8: Computer card is 4 of Clubs; User card is Ace of Diamonds
Computer wins ? user score is 5
==> Enter your card: 32
R9: Computer card is King of Diamonds; User card is 6 of Clubs
Computer wins ? user score is 5
==> Enter your card: 50
R10: Computer card is 9 of Hearts; User card is Jack of Spades
User wins ? user score is 6
End of Game!
```

2. Removed the “static” modifier from the *getInput* method.

Results:

- The program generated a compile error.

```
$ javac WarGame.java
WarGame.java:23: error: non-static method getInput(Scanner) cannot be referenced from a stati
c context
                        user = getInput(scan);
                               ^
1 error
```

Without the “static” modifier, the getInput method needs an instance of the WarGame class to work.