Object Oriented Programming 2021/22
Project self-evaluation form

Oral discussion date _____ Group number ____13_____

Student Number	Student Name	Percentage of participation (must sum 100)	Expected mark (0-10 points)
66325	Tomás Marques Videira Fonseca	33,(3) %	[9; 10]
96135	Afonso Brito Caiado Correia Alemão	33,(3) %	[9; 10]
96317	Rui Pedro Canário Daniel	33,(3) %	[9; 10]

Mark	Command/feature	Correctly implemented	Implemented with faults	Not implemented	Prof notes
UML					
5/20	Tool used: _Visual Paradigm_ Was it done with reverse Engineering?_No_				
Basic game					
1/20	Bet command	X			
1/20	Credit command	X			
1/20	Deal command	X			
1/20	Hold command	X			
Strategy and statistics					
3/20	Perfect strategy	X			
0.5/20	Advice command	X			
0.5/20	Statistics command	X			
Modes					
1/20	Debug mode Reading card and command files and running commands	X			
1/20	Simulation mode Shuffling/Re-shuffling, commands with perfect strategy and statistics	X			
Documentation					
1/20	Examples of debug files Examples to test the game in debug mode	X			
2/20	Java doc Packages, interfaces, classes, methods, fields	X			

Visualization		
1.5/20	Correct in example files without errors A correct output should give a correct info in all commands/advice/statistics	Prof notes:
0.5/20	Correct in example files **with** errors A correct output should not crash and give information to the user	Prof notes:
Discounts		
-3/20	Interfaces and polymorphism used incorrectly	Give here the name of all interfaces in your project: IVideoPokerGame, IHand, ICard, IDeck, IPlayer Give here the name of all abstract classes in your project: Variant, Mode Give here the name of all polymorphic methods in your project: play() in Mode, interface methods and abstract methods that exist in Variant	Prof notes:
-2/20	Open-closed principle used incorrectly	How many packages? 4 Visibility of the attributes (choose all used): ☐ - ~ ☑ # +	Prof notes:
-1/20	Object class / collections used incorrectly	Which classes override equals? Cards, Hand Did you use any sorting method or sorted collection from Java? We used Collections.sort(cardsOrdered, new ComparatorByValueandSuit()); by implementing Comparator interface with class ComparatorByValueandSuit How is the deck shuffled? Collections.shuffle(deck); Did you provide your own exceptions? We used "Exception e" to catch all exceptions in some blocks of code. When we catch an Exception, the program outputs an error message with instructions about it.	Prof notes:
-1/20	Incorrect data structures	Collections used (apart from arrays): List: ArrayList (cards, cardsOrdered and in some local variables) and List: LinkedList (deck, OutOfDeck and in some local variables)	Prof notes:
-1/20	Prints outside the format	Professor notes:
-0.5/20	Problems with the executable Incorrect MANIFEST.MF and JAR w/out java sources	
-0.5/20	Files submitted outside the format Other compression than .zip and incorrect folders	
-2^n/day	Projects submitted after the established date	