Object Oriented Programming 2021/22

Math in Casinos: Video Poker

MEEC/LEEC – IST

1 Game

Most of the material of this section was inspired and adapted from `https://en.wikipedia.org/wiki/Video_poker`. See also `https://en.wikipedia.org/wiki/List_of_poker_hands`.

Video poker is a game based on five-card draw poker played on a computerized console similar in size to a slot machine that made its entry to the casino in the seventies. For the player who likes a game of skill, a low house edge, the possibility of large wins, and the anonymity of playing alone there is nothing else that can compare to video poker. Video poker is an exception to the common adage that the *house always has the advantage*. By choosing the most liberal pay tables, and playing them properly, the player can have a thin advantage.

Video poker rules are as follows:

1. Video poker is played with a standard 52-card deck.
2. After making a wager and pressing the *deal* button the game will **randomly** give the player five cards **from the deck**.
3. The player chooses which cards to discard and which ones to keep.
4. The game replaces the discarded cards with **randomly** chosen cards **from the remaining deck**.
5. The player is paid according to the poker value of his hand (Section 1.1) and the posted paytable (Section 1.3).

1.1 Betting and winning

After inserting money into the machine, play begins by placing a bet of one or more credits and pressing the *deal* button. The player is then given five cards and has the opportunity to discard one or more of them in exchange for new ones; the player may discard all five of their original cards. After the draw, the machine pays out if the hand played matches one of the winning combinations (posted in the paytable, Section 1.3).

Pay tables allocate the payouts for hands and are based on how rare they are and the game variation. A typical pay table starts with a minimum hand of a pair of jacks, which pays even money. Hand combinations are:

Hand	Meaning
Jacks or Better	a pair of Jacks, Queens, Kings or Aces
Two Pair	2 pairs of the same rank
Three of a Kind	3 of the same rank
Straight	a sequence of 5 cards of consecutive value
Flush	any 5 cards of the same suit
Full house	a pair and a three of a kind
Four of a Kind	4 cards of the same value
Straight Flush	5 consecutive cards of the same suit
Royal Flush	a Ten, a Jack, a Queen, a King and an Ace of the same suit

1.2 Variations and full-pay game

There are many variations of video poker. They include *deuces wild*, where a two serves as a wild card; *joker's wild*, where a joker serves as a wild card; pay schedule modification, where four aces with a five or smaller kicker pay an enhanced amount (these games usually have some adjective in the title such as *bonus*, *double*, or *triple*); and multi-play poker, where the player starts with a base hand, and each additional played hand draws from a different set of cards with the base hand (multi-play games are offered in *triple play*, *five play*, *ten play*, *fifty play* and *one hundred play* versions).

When modern video poker games first appeared, the highest-paying common variant of a particular game was called *full-pay*. Game variants that returned a lower payback percentage were termed *short-pay*. Though the term *full-pay* is still in use, today, there are many game variants that return more. Payback percentage expresses the long-term expected value of the player's wager as a percentage if the game is played perfectly. A payback percentage of 99 percent, for example, indicates that for each $100 wagered, in the long run, the player would expect to lose $1 if they played every hand in an optimal way. *Full-pay jacks or better*, for example, offers a payback percentage of 99.54%. Some payback percentages on *full-pay* games are often close to or even in excess of 100 percent.

Casinos do not usually advertise payback percentages, leaving it up to the player to identify which video poker machines offer the best schedules. The payoff schedules for most video poker machines are configured with a pay schedule that pays proportionally more for certain hands (such as a royal flush) when the maximum number of credits (typically 5 coins) is bet. Therefore, players who do not play with the maximum number of credits at a time are playing with a smaller theoretical return.

In this project, we will focus solely on the *double bonus 10/7* variant explained in the next section.

1.3 Double bonus 10/7

Double bonus video poker is a variation of *jacks or better* with a bonus payout for four aces. This variation offers up to a theoretical return of 100.2 percent, when played with perfect strategy — however, this percentage is only on a 10/7 version video poker game (10/7 being the payouts for a full house and a flush). There are many other video poker varieties of 10/6 and 9/6, for

instance, that have slightly lower than the most generous 10/7 version payout. Although the *full-pay* version has a theoretically-positive return, few play well enough to capitalize on it.

Hand	1 credit	2 credits	3 credits	4 credits	5 credits
Royal Flush	250	500	750	1000	4000
Straight Flush	50	100	150	200	250
Four Aces	160	320	480	640	800
Four 2–4	80	160	240	320	400
Four 5–K	50	100	150	200	250
Full House	10	20	30	40	50
Flush	7	14	21	28	35
Straight	5	10	15	20	25
Three of a Kind	3	6	9	12	15
Two Pair	1	2	3	4	5
Jacks or Better	1	2	3	4	5
Theoretical Return	99.1%	99.1%	99.1%	99.1%	100.2%

2 The strategy

Most of the material of this section was inspired and adapted from `https://wizardofodds.com/games/video-poker/strategy/double-bonus/10-7/`. See also `https://wizardofodds.com/play/video-poker/double-bonus/`.

In what follows consider that:

- *High Card* – Jack, Queen, King, or Ace.
- *Outside straight* – An open-ended straight that can be completed at either end, such as the cards 789T.
- *Inside straight* – A straight with a missing inside card, such as the cards 679T. In addition, A234 and JQKA also count as inside straights because they are at an extreme end.
- *Straight flush draw* (type 1) – Straight flush draw in which the number of high cards equals or exceeds the number of gaps (except any ace-low and 234 suited).
- *Straight flush draw* (type 2) – Straight flush draw with one gap, or with two gaps and one high card, or any ace-low, or 234 suited.
- *Straight flush draw* (type 3) – Straight flush draw with two gaps and no high cards.

The following list shows the value of each type of hand on the deal, in order from best to worst. To play a difficult hand, look up all viable ways to play it on the list, and play the highest one. Plays that are not listed, like AK unsuited, should never be played.

1. Straight flush, four of a kind, royal flush
2. 4 to a royal flush
3. Three aces
4. Straight, flush, full house
5. Three of a kind (except aces)

6. 4 to a straight flush
7. Two pair
8. High pair
9. 4 to a flush
10. 3 to a royal flush
11. 4 to an outside straight
12. Low pair
13. AKQJ unsuited
14. 3 to a straight flush (type 1)
15. 4 to an inside straight with 3 high cards
16. QJ suited
17. 3 to a flush with 2 high cards
18. 2 suited high cards
19. 4 to an inside straight with 2 high cards
20. 3 to a straight flush (type 2)
21. 4 to an inside straight with 1 high card
22. KQJ unsuited
23. JT suited
24. QJ unsuited
25. 3 to a flush with 1 high card
26. QT suited
27. 3 to a straight flush (type 3)
28. KQ, KJ unsuited
29. Ace
30. KT suited
31. Jack, Queen or King
32. 4 to an inside straight with no high cards
33. 3 to a flush with no high cards
34. Discard everything

The following list shows all the difficult hands, and how to correctly play them:

1. K♣ Q♣ J♣ T♣ 9♣ – Straight flush or 4 to a royal flush: Keep the straight flush
2. A♢ K♢ Q♠ J♢ T♢ – 4 to a royal flush or Straight: Keep 4 to a royal flush
3. A♠ K♠ J♠ T♠ 9♠ – 4 to a royal flush or Flush: Keep 4 to a royal flush
4. A♡ A♢ A♠ 2♣ 2♠ – Three Aces or Full House: Keep the three aces
5. 4♣ 4♠ 4♡ 5♢ 5♣ – Full House or Three of a Kind (other than aces): Keep the Full House
6. 5♠ 6♠ 7♠ 8♠ J♠ – Flush or 4 to a straight flush: Keep the flush
7. 3♢ 4♡ 5♡ 6♡ 7♡ – Straight or 4 to a straight flush: Keep the straight
8. A♣ K♢ Q♢ J♢ T♠ – Straight or 3 to a royal flush: Keep the straight

9. K♣ Q♣ J♣ 9♣ 4♢ – 4 to a straight flush or 3 to a royal flush: Keep 4 to a straight flush
10. A♡ A♠ K♢ K♠ Q♠ – Two pair or 3 to a royal flush: Keep the two pair
11. J♣ J♢ 4♢ 7♢ 9♢ – High pair or 4 to a flush: Keep High pair
12. Q♠ Q♡ J♡ A♡ 2♣ – High pair or 3 to a royal flush: Keep High pair.
13. 8♣ J♣ Q♣ K♣ 9♡ – 4 to a flush or 3 to a royal flush: Keep the 4 to a flush
14. 2♠ 5♠ 7♠ 9♠ 7♡ – 4 to a flush or Low pair: Keep 4 to a flush
15. T♢ J♢ Q♣ K♢ 5♠ – 3 to a royal flush or 4 to an outside straight: Keep 3 to a royal flush
16. T♡ Q♡ A♡ T♢ 8♣ – 3 to a royal flush or Low pair: Keep 3 to a royal flush
17. 7♣ 7♢ 8♡ 9♠ T♢ – 4 to an outside straight or Low pair: Keep 4 to an outside straight
18. 7♢ 8♢ 9♢ T♠ 4♡ – 4 to an outside straight or 3 to a straight flush (type 1): Keep 4 to an outside straight
19. 7♣ 7♢ 8♢ 9♢ 3♠ – Low pair or 3 to a straight flush (type 1): Keep the low pair
20. K♠ Q♢ J♣ 9♡ 9♣ – Low pair or 4 to an inside straight with 3 high cards: Keep the low pair
21. A♣ K♢ Q♡ J♡ 8♡ – AKQJ unsuited or 3 to a straight flush (type 1): Keep AKQJ unsuited
22. A♣ K♢ Q♡ J♡ 9♠ – AKQJ unsuited or 4 to an inside straight with 3 high cards: Keep AKQJ unsuited
23. A♣ K♢ Q♡ J♡ 2♣ – AKQJ unsuited or QJ suited: Keep AKQJ unsuited
24. 2♡ Q♠ J♠ 9♢ 8♠ – 3 to a straight flush (type 1) or 4 to an inside straight with 2 high cards: Keep 3 to a straight flush (type 1)
25. 8♡ T♡ J♡ 3♠ 5♢ – 3 to a straight flush (type 1) or JT suited: Keep 3 to a straight flush (type 1)
26. K♠ Q♢ J♢ 9♡ 7♣ – 4 to an inside straight with 3 high cards or QJ suited: Keep 4 to an inside straight with 3 high cards
27. A♡ K♣ Q♣ T♢ 6♣ – 4 to an inside straight with 3 high cards or 3 to a flush with 2 high cards: Keep 4 to an inside straight with 3 high cards
28. K♡ Q♠ J♠ 9♢ 3♣ – 4 to an inside straight with 3 high cards or 2 suited high cards: Keep 4 to an inside straight with 3 high cards
29. K♣ Q♡ J♠ 9♡ 8♡ – 4 to an inside straight with 3 high cards or 3 to a straight flush (type 2): Keep 4 to an inside straight with 3 high cards
30. Q♢ J♢ 7♢ 5♡ 4♠ – QJ suited or 3 to a flush with 2 high cards: Keep QJ suited
31. 8♠ 9♡ J♢ Q♢ 2♣ – QJ suited or 4 to an inside straight with 2 high cards: Keep QJ suited
32. Q♠ J♠ 2♡ 3♡ 4♡ – QJ suited or 3 to a straight flush (type 2 or 3): Keep QJ suited
33. K♡ J♡ 3♡ 5♠ 6♣ – 3 to a flush with 2 high cards or 2 suited high cards (except QJ): Keep 3 to a flush with 2 high cards
34. 8♠ 9♣ J♢ Q♢ 4♢ – QJ suited or 3 to a flush with 2 high cards or 4 to an inside straight with 2 high cards: Keep QJ suited
35. 2♣ 3♢ 5♡ J♣ A♣ – 3 to a flush with 2 high cards or 4 to an inside straight with 1 high card: Keep 3 to a flush with 2 high cards

36. K♡ J♡ T♠ 9♢ 6♣ – 2 suited high cards (except QJ) or 4 to an inside straight with 2 high cards: Keep 2 suited high cards
37. A♣ J♣ 2♠ 3♠ 5♠ – 2 suited high cards (except QJ) or 3 to a straight flush (type 2 or 3): Keep 2 suited high cards
38. 8♢ 9♢ J♡ Q♢ 4♣ – 4 to an inside straight with 2 high cards or 3 to a straight flush (type 2 or 3): Keep 4 to an inside straight with 2 high cards
39. 8♠ 9♢ J♢ Q♡ 3♢ – 4 to an inside straight with 2 high cards or 3 to a flush with 1 high card: Keep 4 to an inside straight with 2 high cards
40. 8♣ 9♣ T♢ Q♣ 2♡ – 3 to a straight flush (type 2) or 4 to an inside straight with 1 high card: Keep 3 to a straight flush (type 2)
41. 7♣ 8♡ T♢ J♢ 4♠ – 4 to an inside straight with 1 high card or JT suited: Keep 4 to an inside straight with 1 high card
42. 7♢ 8♣ 9♠ J♢ 2♢ – 4 to an inside straight with 1 high card or 3 to a flush with 1 high card: Keep 4 to an inside straight with 1 high card
43. A♢ 2♣ 4♣ 5♣ 7♠ – 4 to an inside straight with 1 high card or 3 to a straight flush (type 2): Keep 3 to a straight flush (type 2)
44. Q♡ J♠ T♠ 2♢ 4♣ – JT suited or QJ unsuited: Keep JT suited
45. J♡ T♡ 6♡ 7♠ 2♢ – JT suited or 3 to a flush with 1 high card: Keep JT suited
46. J♢ T♢ 2♣ 4♣ 6♣ – JT suited or 3 to a straight flush (type 3): Keep JT suited
47. J♠ T♠ K♣ 3♢ 7♡ – JT suited or KQ, KJ unsuited: Keep JT suited
48. J♣ T♣ A♡ 4♢ 6♠ – JT suited or Ace: Keep JT suited
49. 6♠ 7♣ 8♢ T♡ J♡ – JT suited or 4 to an inside straight with no high cards or 4 to an inside straight with 1 high card: Keep 4 to an inside straight with 1 high card
50. J♢ T♢ 2♣ 5♣ 7♣ – JT suited or 3 to a flush with no high cards: Keep JT suited
51. Q♣ J♡ 9♡ 4♡ 2♠ – QJ unsuited or 3 to a flush with 1 high card: Keep QJ unsuited
52. Q♣ J♢ T♣ 3♠ 5♡ – QJ unsuited or QT suited: Keep QJ unsuited
53. Q♠ J♢ 5♡ 6♡ 9♡ – QJ unsuited or 3 to a straight flush (type 3): Keep QJ unsuited
54. A♢ Q♣ J♠ 4♡ 7♠ – QJ unsuited or Ace: Keep QJ unsuited
55. Q♣ J♡ 2♢ 5♢ 7♢ – QJ unsuited or 3 to a flush with no high cards: Keep QJ unsuited
56. 7♣ T♣ Q♣ 3♡ 2♢ – 3 to a flush with 1 high card or QT suited: Keep 3 to a flush with 1 high card
57. K♠ Q♡ 8♡ 5♡ 2♣ – 3 to a flush with 1 high card or KQ, KJ unsuited: Keep 3 to a flush with 1 high card
58. A♢ 5♢ 9♢ 8♠ 6♡ – 3 to a flush with 1 high card or Ace: Keep 3 to a flush with 1 high card
59. 7♣ T♣ K♣ 3♡ 2♢ – 3 to a flush with 1 high card or KT suited: Keep 3 to a flush with 1 high card
60. 6♣ 7♠ 9♣ T♡ K♣ – 3 to a flush with 1 high card or 4 to an inside straight with no high cards: Keep 3 to a flush with 1 high card
61. Q♡ T♡ 2♣ 4♣ 6♣ – QT suited or 3 to a straight flush (type 3): Keep QT suited
62. K♡ Q♠ T♠ 4♣ 5♢ – QT suited or KQ unsuited: Keep QT suited
63. A♣ Q♢ T♢ 6♠ 9♡ – QT suited or Jack/King/Ace: Keep QT suited

64. Q♢ T♢ 8♡ 7♣ 6♠ – QT suited or 4 to an inside straight with no high cards: Keep QT suited
65. Q♠ T♠ 3♡ 7♡ 8♡ – QT suited or 3 to a flush with no high cards: Keep QT suited
66. 5♣ 6♣ 9♣ K♢ Q♡ – 3 to a straight flush (type 3) or KQ, KJ unsuited: Keep 3 to a straight flush (type 3)
67. 3♡ 5♡ 7♡ J♠ 8♢ – 3 to a straight flush (type 3) or Jack/Queen/King/Ace: Keep 3 to a straight flush (type 3)
68. 2♣ 3♣ 6♣ T♢ K♢ – 3 to a straight flush (type 3) or KT suited: Keep 3 to a straight flush (type 3)
69. 2♢ 3♢ 5♡ 6♢ 9♠ – 3 to a straight flush (type 3) or 4 to an inside straight with no high cards: Keep 3 to a straight flush (type 3)
70. A♢ K♣ J♠ 7♡ 4♣ – KQ, KJ unsuited or Ace: Keep KQ, KJ unsuited
71. K♣ Q♡ T♣ 4♢ 6♠ – KQ, KJ unsuited or KT suited: Keep KQ, KJ unsuited
72. K♡ J♠ 3♢ 8♢ 9♢ – KQ, KJ unsuited or 3 to a flush with no high cards: Keep KQ, KJ unsuited
73. A♣ K♡ T♡ 4♠ 5♢ – Ace or KT suited: Keep Ace
74. A♣ J♡ 5♠ 8♢ 9♣ – Ace or Jack, Queen, King: Keep Ace
75. A♢ 5♡ 6♠ 7♡ 9♢ – Ace or 4 to an inside straight with no high cards: Keep Ace
76. A♠ 2♣ 5♣ 9♣ 6♡ – Ace or 3 to a flush with no high cards: Keep Ace
77. K♢ 6♣ 7♠ 9♡ T♢ – KT suited or 4 to an inside straight with no high cards: Keep KT suited
78. K♣ T♣ 3♡ 5♡ 8♡ – KT suited or 3 to a flush with no high cards: Keep KT suited
79. J♡ 2♠ 3♣ 4♢ 6♡ – Jack/Queen/King or 4 to an inside straight with no high cards: Keep Jack/Queen/King
80. Q♣ 2♡ 5♡ 7♡ 9♠ – Jack/Queen/King or 3 to a flush with no high cards: Keep Jack/Queen/King
81. 2♢ 3♠ 5♣ 6♣ T♣ – 4 to an inside straight with no high cards or 3 to a flush with no high cards: Keep 4 to an inside straight with no high cards

3 Implementation details

The following sections provide further details about project implementation, namely, minimum requirements, program parameters, commands and results, and running in the command line.

3.1 Minimum requirements

In this project, it is required the implementation of the Video Poker Game. The implementation should include the *double bonus* 10/7 variant as described in Section 1.3, with *short-pay* and *full-pay* games (betting 1 to 5), as well as the strategy described in Section 2.

3.2 Program parameters

There are two different modes for playing video poker. A **debug mode**, where the game is fully loaded from a file. A **simulation mode** where the game is automatically played with a perfect strategy to understand the average gain in the player's credit.

3.2.1 Input parameters for debug mode

When loading the commands from a file for debugging purposes the program may receive as input parameters the `credit` with the initial amount of money of the player, the `cmd-file` with the name of the file with the commands, and the `card-file` with the name of the file with the cards (already ordered) for the game (do not shuffle the cards when reading from this file). A few examples of a `cmd-file` will be provided in the "Project section" of the OOP website. Stay tuned!

3.2.2 Input parameters for simulation mode

When performing a simulation the program may receive as input parameters the `credit` with the initial balance of the player, the `bet` with the value to bet during all simulations, and `nbdeals` with the total number on deals. In this case, all games should be played following the perfect strategy described in Section 2 and in the end some statistics should be printed; see "Project section" of the OOP website for examples.

3.3 Commands and results

The possible commands are:

Command	Meaning
`b`	bet
`$`	credit
`d`	deal
`h`	hold
`a`	advice
`s`	statistics

The player must decide how much to bet on a hand before the deal. So, the bet (`b`) command can be used as: `b` or `b` i, where i is the value to bet. If only `b` is typed then: (i) the previous betted amount is used; or `5` is used if there was no previous bet. A `b` command typed after the deal and before the end of the dealer's turn is illegal, and so it should be printed in the terminal ``b: illegal command''. Other commands might be illegal at certain points of the game; in that case, just print a similar warning.

The credit (`$`) can be used at any time of the game. The deal (`d`) can only be used at the beginning of each round, after the bet command. Afterward, the hold (`h`) command can be used. The `h` command prints to the terminal the actual hand of the player, after discarding the cards in exchange for new ones. For example, consider that the player's hand is Q♢ T♢ 8♡ 7♣ 6♠ and he types in the terminal `h 1 2`. In this case, he is holding QT (the cards at index 1 and 2) and discarding 876. Three new cards are drawn to the player and in the end, the player's hand should be printed to the terminal (see examples provided on the Project webpage).

Each time the player loses it should be printed in the terminal ``player loses and his credit is C'', where `C` is the current balance of the player. When the player wins it should be printed in the terminal ``player wins with a H and his credit is C'', where `H` describes the final hand (in capital letters) and `C` is the current balance of the player.

There is also two additional commands available: advice (`a`) and statistics (`s`). The advice prints the next action the player should take, according to the perfect strategy defined in Section 2 (see examples provided on the Project webpage). The `s` command prints to the terminal the average statistics of the game, in the following format:

Hand	Nb
Jacks or Better	N1
Two Pair	N2
Three of a Kind	N3
Straight	N4
Flush	N5
Full house	N6
Four of a Kind	N7
Straight Flush	N8
Royal Flush	N9
Other	N10
Total	N11
Credit	N12 (N13%)

where `N1..N10` is the number of times of the corresponding (final) hand occurred in a deal since the beginning of the game. The `N11` indicates the number of deals since the begging of the game, whereas the `N12` gives the final balance of the player and `N13` the "theoretical returned" as explained in the Double bonus 10/7 table in Section 1.3 (computed as (sum_of_all_gain / sum_of_all_bets) $\times$ 100). Note: the sum of all bets in the simulation mode is just `bet` $\times$ `ndbeals` (see Section 3.4).

In debug mode, the commands are read from a file. All commands described can appear in the `cmd-file` (including `a` even if it is not considered/used!). In addition, the results/prints described for each command should also be printed to the terminal.

If the program is running in simulation mode (and only in this case), in the final of the simulation a table with the statistics must be printed on the terminal. This is the only information required to be printed; that is, do not print the result of each command in the terminal. The commands to use in this case are completely dictated by the perfect strategy described in Section 2.

3.4 Running in the command line

A .jar file must be created so that the program runs by typing in the terminal

```
java -jar <<YOUR-JAR-NAME>>.jar -d credit cmd-file card-file
```

for the debug mode (`-d` option), or by typing in the terminal

```
java -jar <<YOUR-JAR-NAME>>.jar -s credit bet nbdeals
```

for the simulation mode (`-s` option).

4 Grading

The assessment will be based on the following 10-point scale:

1. **(2.5 point)**: UML. The UML will be evaluated, on a 2.5-point scale as: 0–very bad, 1–bad, 1.5–average, 2–good, and 2.5–excellent.

2. **(7.5 points)**: A solution that provides an extensible and reusable framework. The implementation of the requested features in Java is also an important evaluation criterion and the following discounts, on a 5.5-point scale, are pre-established:

(a) **(-2.5 points):** OOP ingredients are not used or they are used incorrectly; this includes polymorphism, open-close principle, etc.

(b) **(-1 points):** Java features are handled incorrectly; this includes incorrect manipulation of methods from `Object`, `Collection`, etc.

(c) **(-0.5 points):** Prints outside the format requested in Section 3.3.

(d) **(-0.5 points):** A non-executable jar file, or a jar file without sources or with sources out of date. Problems in extracting/building a jar file, as well as compiling/running the executable in Java, both from the command line.

Finally, on a 10-point scale:

1. Files submitted outside of the required format will have a penalty of 5% over the respective grade.

2. Projects submitted after the established date will have the following penalty: for each day of delay, there will be a penalty of 2^{n-1} points of the grade, where n is the number of days in delay. That is, projects submitted up to 1 day late will be penalized in $2^0 = 1$ points, incurring a penalty of 0.5 points of the final grade; projects submitted up to 2 days late will be penalized in $2^1 = 2$ points, incurring a penalty of 1 point of the final grade; and so on. Per day of delay, we mean cycles of 24h from the day and hour specified for submission.

5 Deadlines and material for submission

The **deadline for submitting the project is June 20, before 12:00 (with 4 hours of tolerance, that is, after 16:00 penalties start)**. The submission is done via Fenix, so you should ensure that you are registered in a project group.

The following files must be submitted:

1. An UML specification including classes and packages (as detailed as possible), in `.pdf` or `.jpg` format. Place the UML files inside a folder named UML.

2. An executable `.jar` (with the respective source files `.java`, compiled classes `.class`, and `MANIFEST.MF` correctly organized into directories).

3. Five examples of command files (`cmd-file`) and card files (`card-file`) used to test the program in debug mode. Place these examples inside a folder named TESTS.

4. Documentation (generated by the JavaDoc tool) of the application. Place the documentation inside a folder named JDOC.

5. A self-assessment form (in `.pdf` format) that will be made available in due time on the course webpage. Place the self-assessment form inside the folder named DOCS.

The UML folder, executable (the `.jar` file with the source files, besides the compiled files and `MANIFEST.MF`), the JDOC folder and the DOCS folder, should be **submitted via fenix in a single `.zip` file**.

The final discussion will be held from Jun 22 to June 24. The distribution of the groups for final discussion will be available in due time. All group members must be present during the discussion. **The final grade of the project will depend on this discussion, and it will be not necessarily the same for all group members.**