2016-02-17

4 decks of cards : 1 → 10, J (11), Q (12), K (13)

3 questions → question correct? YES ⇒ TOKEN, NO ⇒ nothing

token → draw one card or not

First, draw all the possible / wanted cards

BASIC RULE (≥ 1° year) : 6 + sum of points of rds (ITEM 1)

ADDENDUM RULE (≥ 2° year) : J, Q, K → 10 points (ITEM 2)

EXTRA RULE (≥ 3° year) : ACE → 14 (insead of 1) (ITEM 3)

SUPER RULE (4° and 5° years) : choise : (ITEM 4)

* pointsOfCards + 6

OR * 52 - pointsOfCards

```
DECKS   DB   4*52 DUP(?)        ← input array (initialized within the code)
  └→ Entry:    XXSSVVVV
               |  |  └→ V = Value of the card [1-13]
               |  └→ S = suit (ØØ = SPADES, Ø1 = HEARTHS, 1Ø = DIAMONDS, 11 = CLUBS)
               └→ NOT USED (ØØ)
```

```
STUDENTS   DB   ?          ; number of students evaluated (to check max = 30)
TOKENS     DB   ?          ; #tokens owned by the current student
SENIORITY  DB   ?          ; attendance year of the current student
VOTE       DB   ?          ; final vote of the current student
CARDS      DB   3 DUP(Ø)   ; value of the cards of the current student
```

```
1.  ASK FOR #TOKENS → STORE IN "TOKENS" variable
2.  ASK FOR ATTENDANCE YEAR → STORE IN "SENIORITY" variable
3.  DRAW CARDS :
        a.  show card   ( value and suit)
        b.  store value in CARDS [SI]     ; SI incremented from Ø to 2
        c.  other available tokens?
              YES → d. , NO → 4. compute vote
        d.  ask if the student wants to draw a new card
              YES → 3 , NO → 4. compute vote
4.  COMPUTE VOTE
        a.  if SENIORITY = 1 ⇒ g. compute Basic
        b.  ask for ADDENDUM RULE
              ↳ YES → call addendumRuleProc , NO → c.
        c.  if SENIORITY ≥ 3 → d.ask for EXTRA RULE
              otherwise → g. compute Basic
        d.  ask for EXTRA RULE
              ↳ YES → call extraRuleProc, NO → e.
        e.  if SENIORITY ≥ 4 → f. ask for SUPER RULE
              otherwise     compute Basic
        f.  Select   [1] 6 + points of cards ⇒ g. compute Basic
                     [2] 52 - points of cards ⇒ h. compute super
        g.  COMPUTE BASIC :
                -   sum points of drawn cards + 6
                -   jmp 5. SHOW VOTE
        h.  COMPUTE SUPER :
                -   52 - points of drawn cards
                -   jmp 5.SHOW VOTE
```

5. SHOW VOTE:

   a. print final vote + rejected / exam passed

   b. if STUDENTS < 30 ⟹ ask for next student
      otherwise end of program

Example of DECKS array:

00111101	(K CLUBS)
00000100	(4 SPADES)
00111100	(Q CLUBS)
00010010	(2 HEARTHS)
00011101	(K HEARTHS)
00101011	(J DIAMONDS)
00000001	(1 SPADES)
00100001	(1 DIAMONDS)
00001000	(8 SPADES)

Example of execution of the program:

TOKENS	SENIORITY	OMP	CARD 1	CARD 2	CARD 3	ADDENDUM	EXTRA	SÜPER	VOTE
3	1	NO	K CLUBS	4 SPADES	X	X	X	X	23
3	2	NO	Q CLUBS	2 HEARTHS	K HEARTHS	YES	X	X	28
2	3	YES	J DIAM.	1 SPADES	X	YES	YES	X	30
2	4	YES	1 DIAM.	8 SPADES	X	NO	YES	SUBTRACT	30

```
; Procedure to apply the ADDENDUM role
        PUSH BP
        MOV BP, SP
        ; push all used registers (AX, CX, SI)
        MOV SI, [BP + 4]   ; base address of CARDS array
        MOV CX, 3
        MOV AL, 10
resizeTen:
        CMP [SI], AL        ; if value ≤ 10 => do nothing ...
        JLE nextCardTen
        MOV [SI], AL        ; ... otherwise set it to 10
nextCardTen:
        add SI, 1
        loop resizeTen
        ; pop all used registers --> end of procedure


; Procedure to apply EXTRA role
        PUSH BP
        MOV BP, SP
        ; push all used registers (AX, CX, SI)
        MOV SI, [BP+4]
        MOV CX, 3
        MOV AL, 14
resizeOne:
        MOV AH, 1
        CMP [SI], AH
        JNE nextCardOne      ; if value != 1 => do nothing...
        MOV [SI], AL         ; ...otherwise set it to 14
nextCardOne:
        ADD SI, 1
        loop resizeOne
        ; pop all used registers --> end of procedure
```