{ADRIAN PEÑALVER FERNANDEZ}

{PEC\_4}

{01/11/2022}

{ALGORITMO DE WHILE Y FOR}

**const**

MIN\_CINEMAS: **integer** = 1; {Min. number of cinemas}

MAX\_CINEMAS: **integer** = 10; {Max. number of cinemas}

SCREENS\_3: **integer** = 3; {3 screens}

SCREENS\_6: **integer** = 6; {6 screens}

POINTS\_LE\_3\_SCREENS: **real** = 2.0; {Points less/equals 3 screens}

POINTS\_LE\_6\_SCREENS: **real** = 3.0; {Points less/equals 6 screens}

POINTS\_GT\_6\_SCREENS: **real** = 4.0; {Points greater than 6 screens}

PARKING\_POINTS\_INCREASE: **real =** 1.5**;** {Points increase if has parking}

**end const**

**type**

tCinemaType = {PREMIERE, RERELEASE, INDIE, OTHER}

**end type**

**algorithm** UOCCinema

{Variable definitions}

**var**

cinemaIdVector: **vector**[MAX\_CINEMAS] **of integer**;

cinemaTypeVector: **vector**[MAX\_CINEMAS] **of** tCinemaType;

numScreensVector: **vector**[MAX\_CINEMAS] **of integer**;

hasParkingVector: **vector**[MAX\_CINEMAS] **of boolean**;

totalPointsVector: **vector**[MAX\_CINEMAS] **of real**;

numCinemas: **integer**;

bestCinema: **integer**;

maxPoint: **real;**

**end var**

{Exercise 2.1}

{Data input}

**writeString**("INPUT DATA");

**writeString**("NUMBER OF CINEMAS (1-10)? >>");

numCinemas:= **readInteger**();

{Data validation}

**While** MIN\_CINEMAS > numCinemas or numCinemas > MAX\_CINEMAS **do**

**writeString**("INVALID DATA, TRY AGAIN!”);

**writeString**("NUMBER OF CINEMAS (1-10)? >>");

numCinemas:= **readInteger**();

**End while**

{Exercise 2.2}

maxPoint:= 0;

**for** i:= 1 **to** numCinemas **do**

{Data input}

**writeString**("CINEMA #");

**writeInteger**(i+1);

**writeString**("ID (AN INTEGER)? >>");

cinemaIdVector[i]:= **readInteger**();

**writeString**("TYPE (1-PREMIERE, 2-RERELEASE, 3-INDIE, 4-OTHER)? >>");  
cinemaType:= **readCinemaType**();

**writeString**("SCREENS (1-10)? >>");

numScreensVector[i]:= **readInteger**();

**writeString**("HAS PARKING (0-FALSE, 1-TRUE)? >>");

hasParkingVector[i]:= **readBoolean**();

{Data processing}

**If** numScreensVector[i]<= SCREENS\_3 **then**

totalPointsVector[i]:= POINTS\_LE\_3\_SCREENS;

**else**

**if** SCREENS\_3 < numScreensVector[i] and numScreensVector[i] <= SCREENS\_6 **then**

totalPointsVector[i]:= POINTS\_LE\_6\_SCREENS;

**else**

totalPointsVector[i]:= POINTS\_GT\_6\_SCREENS;

**end if**

**end if**

**if** hasParkingVector[i] **then**

totalPointsVector[i]:= totalPointsVector[i]\* PARKING\_POINTS\_INCREASE;

**end if**

**if** totalPointsVector[i] > maxPoint **then**

maxPoint:= totalPointsVector[i];

bestcinema:= i;

**end if**

**end for**

{Data output}

{Exercise 2.3}

**writeString**("RESULTS");

**writeString**("BEST CINEMA [ID]:");

**writeInteger**( cinemaIdVector[bestCinema]);

**writeString**("TOTAL POINTS:");

**writeReal**( totalPointsVector[bestCinema]);

**writeString**("SCREENS:");

**writeInteger**( numScreensVector[bestCinema]);

**writeString**("TYPE (1-PREMIERE, 2-RELEASE, 3-INDIE, 4-OTHER): ");

**WriteCinemaType**(cinemaType);

**writeString**("HAS PARKING (0-FALSE, 1-TRUE): ");

**writeBoolean**(hasParkingVector[bestCinema]);

**end algorithm**