PAC1 – DANIEL GALEANO SANCHO

Format i data de lliurament

La PAC cal lliurar-se abans del dia **9 de Març de 2020 a les 23:59**. Per al lliurament caldrà que entregueu un fitxer en format **ZIP**, que contingui:

- Un fitxer **pdf** amb les respostes als diferents exercicis que es plantegen.
- Els fitxers **chess.h** i **chess.c** demanats.

Cal fer el lliurament a l'apartat de lliuraments d'AC de l'aula de teoria.

Presentació

En aquesta PAC treballarem els conceptes bàsics de programació que es donen per assolits en l'assignatura prèvia, i que són la base d'aquesta assignatura. A partir d'aquest punt inicial, afegirem l'especificació formal dels algoritmes i la seva traducció al llenguatge C. Finalment, treballarem el concepte de disseny descendent. A diferència de Fonaments de Programació, en que els problemes es donaven molt pautats, s'espera que en aquesta assignatura sigueu capaços de resoldre els problemes a partir de la seva descripció en llenguatge natural.

Competències

Transversals

- Capacitat de comunicació en llengua estrangera.

Específiques

- Capacitat de dissenyar i construir aplicacions informàtiques mitjançant tècniques de desenvolupament, integració i reutilització.
- Coneixements bàsics sobre l'ús i la programació dels ordinadors, sistemes operatius, bases de dades i programes informàtics amb aplicació a la enginyeria

Objectius

- Saber realitzar petits programes per solucionar problemes a partir d'una descripció general del problema.
- Saber especificar un algoritme mitjançant les pre i post condicions.
- Saber incloure controls en els programes per tal de garantir que les pre condicions es donen.
- Saber reduir un problema donat en problemes més petits mitjançant el disseny descendent.

Recursos

Per realitzar aquesta activitat teniu a la vostra disposició els recursos:

Bàsics

- Materials en format web de l'assignatura de Fonaments de Programació.
- Materials en format web de l'assignatura de les 3 primeres setmanes.
- Laboratori de C.

Complementaris

- Internet: La forma més efectiva de trobar informació sobre qualsevol dubte sobre C és la cerca a través d'un cercador.
- La solució de la PAC d'un semestre anterior.

Criteris de valoració

Cada exercici porta associat la puntuació sobre el total de l'activitat. Es valorarà tant la correctesa de les respostes com la seva completesa.

- En exercicis on es demana llenguatge algorísmic, cal respectar el format.
- En el cas d'exercicis en llenguatge C, aquests han de compilar per ser avaluats. En tal cas, es valorarà:
  - Que funcionin
  - Que es respectin els criteris d'estil i que el codi estigui comentat
  - Que les estructures utilitzades siguin les correctes

[20%] Exercici 1: Definició de tipus de dades

Un torneig d'escacs vol informatitzar el seu sistema de rondes i emparellaments entre jugadors. A continuació, s'enumeren els detalls del sistema que es vol crear:

- Al torneig es poden inscriure un nombre indefinit de jugadors, dels quals es necessita saber: identificador (enter major que 0), nom, edat i nacionalitat. A més, de cada jugador també ens interessa emmagatzemar la seva puntuació personal "elo" (enter major que 0) i el seu nivell. Segons sigui el seu elo, tindrà un determinat nivell: si l'elo està entre 0 i 1800, el nivell serà "novice"; si l'elo està entre 1801 i 2000, el nivell serà "medium"; si l'elo està entre 2001 i 2400, el nivell serà "advanced"; i si l'elo és major de 2400, el nivell serà "màster".

- Per a cada emparellament (és a dir, per a cada partida disputada entre dos jugadors) ens interessa saber l'identificador de l'emparellament, l'identificador del jugador que condueix les peces blanques, l'identificador del jugador que condueix les peces negres i el resultat de la partida. Cada partida jugada té 3 resultats possibles: guanya el jugador de peces blanques (whiteWins), guanya el jugador de peces negres (blackWins) o es produeix un empat (draw).

- Cada ronda del torneig està formada per un màxim de 5 emparellaments (és a dir, 10 jugadors). De cada ronda ens interessa saber el número de ronda (les rondes van numerades consecutivament començant pel 1), els emparellaments, el nombre d'emparellaments, el nombre de victòries que s'han produït amb peces blanques, el nombre de victòries amb peces negres i el nombre d'empats.

- Del torneig necessitem saber el seu nom, la seva localització, la data en què es disputa (es pressuposa que es juga sencer en un dia), el llistat de jugadors inscrits, el nombre de jugadors inscrits i la informació de totes les rondes disputades (que seran 9 com a màxim), a més del nombre de rondes disputades.

A partir d'aquest enunciat, es demana (**en llenguatge algorísmic**):

a) Definir un tipus de dades **tPlayer** que emmagatzemi la informació d'un jugador inscrit en el torneig.

b) Definir un tipus de dades **tPairing** que guardi la informació d'un emparellament.

c) Definir un tipus de dades **tRound** que representi la informació d'una ronda.

d) Finalment, definir un tipus de dades **tChessTournament** que contingui tota la informació del torneig.

Nota: Podeu definir els tipus addicionals que considereu oportuns.

```
const
        MAX_PAIRINGS : integer := 5;
        MAX_PLAYERS : integer := 10;
        MAX_ROUNDS : integer := 9;
end const
type
        tPlayer = record
                idPlayer : integer;
                name : string;
                age : integer;
                nationality : string;
                elo : integer;
                level : string;
        end record

        tResult = {draw, whiteWins, blackWins};

        tPairing = record
                idPairing : integer;
                idWhitePlayer : integer;
                idBlackPlayer : integer;
                result : tResult;
        end record
        tRound = record
                idRound : integer;
                pairings : vector[MAX_PAIRINGS] of tPairing;
                numPairings : integer;
                numWhiteWins : integer;
                numBlackWins : integer;
                numDraws : integer;
```

```
    end record

    tDate = record
        day : integer;
        month : integer;
        year : integer;
    end record

    tChessTournament = record
        name : string;
        location : string;
        date : tDate;
        players : pointer to tPlayer;
        numPlayers : integer;
        rounds : vector[MAX_ROUNDS] of tRound
        numRounds : integer;
    end record
end type
```

[40%] Exercici 2: Manipulació de taules

A partir de les estructures de dades definides en l'exercici 1, defineix els següents mètodes (accions o funcions) **en llenguatge C** (utilitza els fitxers **chess.h** i **chess.c** per declarar i implementar els mètodes):

a) **init_chess_tournament [5%]:** Donada una estructura de tipus **tChessTournament**, la inicialitza amb els següents camps: nom "OPEN CHESS 2020", localització "SPAIN", data "30/1/2020". El nombre de jugadors i el nombre de rondes s'inicialitzen a 0..

b) **new_player [10%]:** Donada una estructura de tipus **tChessTournament**, afegeix a la llista de jugadors un nou jugador amb el seu identificador, nom, edat, nacionalitat, elo i nivell. El nivell ha d'afegir-se en funció del seu elo (per a calcular i assignar el nivell al jugador, es recomana implementar una acció auxiliar). L'acció ha de

comprovar si ja existeix l'identificador del jugador en el torneig, i en aquest cas no s'afegirà i es mostrarà un missatge d'error.

c) **new_round [5%]:** Donada una estructura de tipus **tChessTournament** i un número de ronda, afegeix una ronda al torneig amb aquest número. La resta de camps interns els inicialitza a 0. A més, incrementa en una unitat el comptador de nombre de rondes de **tChessTournament**.

d) **add_pairing [10%]:** Donada una estructura de tipus **tChessTournament**, un identificador d'emparellament, els identificadors de dos jugadors i un número de ronda, afegeix en aquesta ronda un emparellament format pels dos jugadors i s'incrementa en una unitat el nombre d'emparellaments d'aquesta ronda. El primer jugador passat per paràmetre jugarà amb les peces blanques i el segon amb negres. Si algun dels dos jugadors no està en la llista de jugadors, no s'afegirà l'emparellament i es mostrarà un error. Si algun dels dos jugadors ja ha estat emparellat en aquesta ronda, no s'afegirà l'emparellament i es mostrarà un error. Si s'ha aconseguit el nombre màxim d'emparellaments per a aquesta ronda, no s'afegirà l'emparellament i es mostrarà un error. Si el número de ronda és major que 9, no s'afegirà l'emparellament i es mostrarà un missatge d'error. Per a assignar un resultat a l'emparellament s'utilitzarà la funció rand(). Concretament, si (rand()%3+1 == 1) assignarem la victòria a les blanques, si (rand()%3+1 == 2) assignarem la victòria a les negres, i si (rand()%3+1 == 3), declararem un empat. En funció del resultat, s'incrementarà el nombre de victòries amb peces blanques, amb negres o empats en aquesta ronda.

e) **print_round_pairings [10%]:** Donada una estructura de tipus **tChessTournament** i un número de ronda, mostra per pantalla una capçalera amb el nom, localització, data del torneig i el nombre de jugadors inscrits. A més, per a aquesta ronda, mostra el nombre de partides que s'han guanyat amb blanques, amb negres i el nombre d'empats. Finalment, mostra tots els emparellaments per a aquesta ronda amb el seu identificador d'emparellament, nom, edat, nacionalitat i elo del jugador amb blanques i del jugador amb negres i el resultat de la partida.

Heu d'utilitzar la rutina principal que se us facilita al fitxer **main.c sense cap modificació**. Aquesta rutina realitza les següents tasques::

a. Inicialitza el torneig amb les dades descrites anteriorment, sense jugadors i sense rondes.

b. Afegeix 12 jugadors al torneig amb les seves respectives dades

c. Afegeix al torneig 9 rondes buides numerades de l'1 al 9.

d. Afegeix 5 emparellaments en la ronda 1 i 3 emparellaments en la ronda 2.

e. Imprimeix per pantalla la informació del torneig per a les rondes 1 i 2.

Per considerar que el programa funciona correctament, el resultat final per pantalla haurà de ser el següent (les victòries amb peces blanques, negres i empats podrien variar atès que es calculen amb la funció de números aleatoris rand()):

```
ERROR: Player 12345 is already in the list of players
ERROR while pairing player 98765. He/she does not appear in the list of players
ERROR while pairing player 90123. He/she is already paired in round 1
ERROR while adding the pairing 7 to round 1. It is already full of pairings
**********************************************************************
OPEN CHESS 2020 SPAIN 30/1/2020 (12 players in the tournament)
**********************************************************************
ROUND 1
----------------------------------------------------------------------
Number of pairings: 5
Number of wins with white pieces: 2
Number of wins with black pieces: 2
Number of draws: 1
Pairing id: 1
White: Magnus Carlsen, 29 years old, from Norway, 2862 elo
Black: Fabiano Caruana, 27 years old, from USA, 2354 elo
Result: Black wins
Pairing id: 2
White: Ding Liren, 25 years old, from China, 2551 elo
Black: Sabrina Vega, 37 years old, from Spain, 2350 elo
Result: White wins
Pairing id: 3
White: Francisco Vallejo, 38 years old, from Spain, 2701 elo
Black: Vladimir Kramnik, 44 years old, from Russia, 2152 elo
Result: Draw
Pairing id: 4
White: Vishy Anand, 48 years old, from India, 2425 elo
Black: Judit Polgar, 29 years old, from Hungary, 2605 elo
Result: Black wins
Pairing id: 5
White: David Anton, 22 years old, from Spain, 2664 elo
```

```
Black: Wesley So, 18 years old, from Philippines, 1650 elo
Result: White wins
**********************************************************************
OPEN CHESS 2020 SPAIN 30/1/2020 (12 players in the tournament)
**********************************************************************
ROUND 2
----------------------------------------------------------------------
Number of pairings: 3
Number of wins with white pieces: 1
Number of wins with black pieces: 0
Number of draws: 2
Pairing id: 1
White: Francisco Vallejo, 38 years old, from Spain, 2701 elo
Black: Vishy Anand, 48 years old, from India, 2425 elo
Result: White wins
Pairing id: 2
White: Vladimir Kramnik, 44 years old, from Russia, 2152 elo
Black: Judit Polgar, 29 years old, from Hungary, 2605 elo
Result: Draw
Pairing id: 3
White: David Anton, 22 years old, from Spain, 2664 elo
Black: Fabiano Caruana, 27 years old, from USA, 2354 elo
Result: Draw
**********************************************************************
```

[20%] Exercici 3: Especificació formal

Defineix les declaracions (**no les implementacions**) dels mètodes **init_chess_tournament** i **new_player** de l'exercici anterior en llenguatge algorísmic.

a) Per cada declaració, afegeix les pre i post condicions en llenguatge algorísmic.

b) Afegeix al codi en llenguatge C de l'exercici anterior, els "asserts" necessaris per assegurar que es compleixen les pre-condicions especificades.

a) **action** init_chess_tournament (**in/out** c : tChessTournament)

Pre: {c = C}
No cal definir variables doncs totes venen com paràmetres

Post: {c.name = "OPEN CHESS 2020" ∧ c.location = "SPAIN" ∧ c.date.day = 30 ∧ c.date.month = 1 ∧ c.date.year = 2020 ∧ c.numPlayers = 0 ∧ c.numRounds = 0}

b) **action** new_player (**in/out** c : tChessTournament, **in** idPlayer : **integer**, **in** name : **string**, **in** age : **integer**, **in** nationality : **string**, **in** elo : **integer**)

Pre: {c = C ∧ idPlayer = IDPLAYER ∧ IDPLAYER > 0 ∧ name = NAME ∧ age = AGE ∧ nationality = NATIONALITY ∧ elo = ELO ∧ ELO > 0}
No cal definir variables doncs totes venen com paràmetres

Post: { (i : 0 < i ∃i : 0 < i ≤ C.numPlayers : c.numPlayers = C.numPlayers ∧ c.player[i].idPlayers = IDPLAYER) v
(c.numPlayers = C.numPlayers + 1 ∧
c.players[c.numPlayers].idPlayer = IDPLAYER ∧
c.players[c.numPlayers].name = NAME ∧
c.players[c.numPlayers].age = AGE ∧
c.players[c.numPlayers].nationality = NATIONALITY ∧
c.players[c.numPlayers].elo = ELO ∧
((c.players[c.numPlayers].level = "novice" ∧ c.players[c.numPlayers].elo ≤ 1800) v
(c.players[c.numPlayers].level = "medium" ∧ c.players[c.numPlayers].elo ≤ 2000) v
(c.players[c.numPlayers].level = "advanced" ∧ c.players[c.numPlayers].elo ≤ 2400) v

(c.players[c.numPlayers].level = “master” ∧ c.players[c.numPlayers].elo > 2400))
) }

[20%] Exercici 4: Disseny descendent

Defineix **en llenguatge algorísmic** l'acció **levels_winners** que mostra un llistat per pantalla amb els nivells que tenen els jugadors que han guanyat en cada ronda.

action levels_winners (c: tChessTournament)

Per implementar aquesta funció s'ha de descompondre en accions o funcions més simples (mitjançant el disseny descendent) que també s'hauran d'implementar.

Els tipus que s'utilitzen estan definits a l'exercici 1.

Nivell 1:

```
action level_winners (in c: tChessTournament)
        var
                idRound : integer;
        end var

        idRound := 1;

        while idRound < c.numRounds do
                writeString(“Round: ”);
                writeInteger(idRound);
                print_winner_level(c, idRound);
                idRound := idRound + 1;
        end while
end action
```

Nivell 2:

```
action print_winner_level (in c: tChessTournament, in idRound : integer)
        var
                posPairing : integer;
                level : string;
        end var

        posPairing := 0;

        while posPairing < c.rounds[idRound].numPairings do
                level := get_level_winner(c, idRound, posPairing);
                if level ≠ "nul" then
                        writeString(level);
                end if
                posPairing := posPairing + 1;
        end while
end action
```

Nivell 3:

```
function get_level_winner (in c: tChessTournament, in idRound : integer, in
posPairing : integer) : string
        var
                level : string;
                result : integer
        end var

        string level := "nul";

        result := c.rounds[idRound].pairings[posPairing].result;

        if result = whiteWins then
                level := find_player(c,c.rounds[idRound].
                pairings[posPairing].idWhitePlayer)->level;
        else
                if result = blackWins then
                        level := find_player(c,c.rounds[idRound].
                        pairings[posPairing].idBlackPlayer)->level;
                end if
        end if
        return level;
end function
```

Nivell 4:

```
action find_player (in c: tChessTournament, in idPlayer : integer)
        var
                res : tPlayer;
                i : integer;
        end var

        i := 0;
        res := NULL;
        while i < c.numPlayers && res = NULL do
                if c.players[i].idPlayer = idPlayer then
                        res := c.players[i];
                else
                        i := i +1;
                end if
        end while

        return res;
end action
```