Examen de laboratori d'EC

25 de maig de 2021

INSTRUCCIONS IMPORTANTS:

Baixa els fitxers plantilla de la web d'exàmens

1. La imatge Linux que cal usar hauria d'estar precarregada ja quan comencis a fer l'examen. Si no ho està, cal usar la imatge "**Linux**" que apareix al menú de possibles imatges.
2. Entra al teu compte personal amb el teu nom d'usuari d'estudiant.
3. Obre el navegador a la pàgina: `https://examens.fib.upc.edu`
4. Clica *Accedir a l'aplicació d'exàmens*, i autentifica't de nou amb el teu nom d'usuari d'estudiant.
5. Busca la pràctica *Examen de Laboratori d'EC - Quadrimestre primavera 2020/21 – Subgrup 21*
6. Obre el fitxer adjunt **examlab-2021q2-subgrup21.zip**, i extreu els fitxers a la teva carpeta. Comença l'examen.

Resol l'examen

1. Posa nom i cognoms a la capçalera dels fitxers següents i que s'han de lliurar de forma individual:
   matrius.s, **subrutines.s, cache_respostes.txt**
2. No canvïis el nom dels fitxers.
3. Pots pujar cada fitxer a la web múltiples cops, no esperis a l'últim minut.
4. L'avaluació de cada apartat tindrà en compte principalment el seu funcionament correcte. Es practicaran diverses comprovacions, a més a més de la que suggereix el propi enunciat.
5. Per executar el simulador escriviu la següent comanda en una finestra de terminal:

```
java -jar /assig/ec/Mars/Mars.jar &
```

Puja les solucions a la web d'exàmens

1. Busca al final de la pàgina web el quadre *Lliurar una nova pràctica*, i verifica que la llista desplegable té seleccionada la pràctica: *Examen de laboratori d'EC - Quadrimestre primavera 2020/21 – Subgrup 21*
2. Clica el botó *Browse* i puja d'un en un els fitxers a lliurar amb la solució:
   matrius.s, **subrutines.s, cache_respostes.txt**
3. Verifica que cada fitxer que has pujat no és un fitxer plantilla buit (comprovant la mida en bytes!) i que els noms són els corresponents a aquest torn d'examen.
4. Surt de la sessió que tinguis oberta però **NO APAGUIS EL PC**.

Problema 1. (3,5 punts)

El següent codi en llenguatge C defineix una matrix de mida 4x6 i troba el màxim valor absolut d'entre tots els seus elements **positius**:

```
int matrix[4][6] = {{ 4,  1,  3,  5,  2,  2},
                    { 0,  3,  0,  5,  0,  0},
                    { 2,  2,  1,  9,  1,  0},
                    { 0,  1,  1,  1,  1,  0}};

void main() {
    int max_abs = 0;
    int i = 0;

    for (i=0; i<4; i++) {
        for (int j = 0; j < 6; ++j) {
            if ( matrix[i][j] > max_abs )
                max_abs = matrix[i][j];
        }
    }
}
```

- Traduïu al fitxer **matrius.s** el codi de *main( )* en assemblador MIPS. Les variables globals ja estan declarades i incialitzades.
- Feu servir la crida syscall per imprimir (mostrar per pantalla) el resultat *max abs*
- Comproveu que el codi és també correcte quan la matriu conté elements amb valor negatiu
- Minimitzeu, si és possible, el número d'instruccions i de registres emprats.

Problema 2. (3,5 punts)

El següent codi en llenguatge C ordena, de major a menor, els elements del vector *V[ ]*:

```
int V[10] = {2, 3, 5, -1, 1, 8, 6, 7, 1, -2};

void canvia (int *a, int *b) {
  int temp = *a;
  *a=*b;
  *b=temp;
}

void subr (int vect[], int i, int n) {
  for (int j=i+1; j<n; j++) {
    if (vect[j] > vect[i]) {
      canvia (&vect[j], &vect[i]);
    }
  }
}

void main() {
  int = 0;

  for (int i=0; i<9; i++) {
    ordena(V, i, 10);
  }
}
```

- Traduïu la subrutina *subr( )* a codi assemblador MIPS. Feu-ho al fitxer **subrutines.s**, on ja hi trobareu la funció principal *main( )* i la subrutina *canvia( )* implementades.

Problema 3. (3 punts)

Considereu un sistema computador format per un processador MIPS, una memòria principal (MP), una memòria cau o *cache* de dades (MC) amb les següents característiques:

- Capacitat total: 4 blocs
- Mida del bloc: 16 bytes (4 words)
- Correspondència directa
- Escriptura immediata amb assignació

A partir del següent codi d'alt nivell en llenguatge C (en trobareu l'equivalent en assemblador MIPS al fitxer **cache.s**):

```
int vect[8];
int matx[8][8];

void main() {
    int i, temp;     /* En desen en registres */

    for (i = 0; i < 7 i++) {
        for (int j = 0; j < 8; j++) {
            temp = matx[i][j];
            V[j] = V[j] - temp;
        }
    }
}
```

i considerant que les variables globals del programa s'emmagatzemen en memòria a partir de l'adreça 0x10010000 i que la MC es troba buida inicialment, responeu al fitxer **cache_respostes.txt**:

a) Quin percentatge de fallades s'ha produït a la MC quan s'executa tot el codi? (0,4 punts)
b) Algunes iteracions del bucle for extern tenen més fallades. Quines són? A què és degut això? Expliqueu què causa aquesta diferència de comportament. (0,5 punts)
c) De vegades, canviar el grau d'associativitat de la cache (mantenint la mida de bloc i la capacitat total) aporta un benefici en el nombre d'encerts i fallades. És el cas d'aquest programa? Justifiqueu-ho. Justifiqueu-ho. (0,7 punts)
d) Quin és el mínim nombre de fallades que podeu aconseguir només canviant el grau d'associativitat? Quin és el grau d'associativitat més petit que ho aconsegueix? Quin és el guany en la taxa d'encerts respecte al comportament previ? (0,7 punts)
e) Quin és el mínim nombre de fallades que podeu aconseguir canviant el grau d'associativitat i la mida de bloc, mantenint la mida total de la MC? Quin és el guany en la taxa d'encerts respecte al comportament previ? Expliqueu perquè canviar la mida del bloc (i el grau l'associativitat) afecta al nombre de fallades en aquest programa i quin fenomen aprofiteu. (0,7 punts)