2º Grado Informática
Estructura de Computadores
8 Febrero 2016

Nombre:	
DNI:	**Grupo:**

Test de Prácticas (4.0p)

Todas las preguntas son de elección simple sobre 4 alternativas.
Cada respuesta vale 4/20 si es correcta, 0 si está en blanco o claramente tachada, -4/60 si es errónea.
Anotar las respuestas (a, b, c o d) en la siguiente tabla.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20

1. La dirección efectiva del primer parámetro de llamada a una función suele calcularse desde el código de la función como:

a. EBP+8
b. EBP-8
c. EBP+4
d. EBP-4

2. El comienzo de un procedimiento que siga la convención cdecl es:

a. mov %ebp,%esp; push %ebp
b. mov %esp,%ebp; push %ebp
c. push %ebp; mov %ebp,%esp
d. push %ebp; mov %esp,%ebp

3. Considere una función C declarada así:

```
void fun4arg (int a,int b,int c,int d);
```

Suponiendo que fun4arg se ha compilado para una máquina x86 IA-32 con enteros de 4 bytes, ¿cuál sería la dirección del argumento b relativa a %ebp, en el marco de pila de fun4arg?

a. %ebp + 8
b. %ebp + 12
c. %ebp + 16
d. %ebp + 20

4. ¿Cuál de las siguientes afirmaciones sobre las caches es ***FALSA***?

a. Casi ningún procesador actual tiene memoria cache L2
b. Las direcciones a las que accede un programa no son completamente aleatorias, sino que se rigen por ciertos patrones de localidad
c. Un procesador actual tiene varias cachés de nivel 1
d. La caché de nivel 3 no contiene toda la memoria que maneja el programa

5. En un sistema Linux x86-64, ¿cuál de las siguientes variables ocupa más bytes en memoria?

a. char a[7]
b. short b[3]
c. int *c
d. float d

6. En la práctica "suma" se pide sumar una lista de 32 enteros SIN signo de 32bits en una plataforma de 32bits sin perder precisión, esto es, evitando acarreos. ¿Cuál es el mínimo valor entero que repetido en toda la lista causaría acarreo con 32bits (sin signo)?

a. `0xfc00 0000`
b. `0xfbff ffff`
c. `0x0800 0000`
d. `0x07ff ffff`

7. En la práctica "suma" se pide sumar una lista de 32 enteros CON signo de 32bits en una plataforma de 32bits sin perder precisión, esto es, evitando

desbordamiento. ¿Cuál es el valor negativo más pequeño (en valor absoluto) que repetido en toda la lista causaría desbordamiento con 32bits (en complemento a 2)?

a. `0xfc00 0000`
b. `0xfbff ffff`
c. `0xf800 0000`
d. `0xf800 0001`

---

8. ¿Qué valor contendrá edx tras ejecutar las siguientes instrucciones?

```
xor %eax, %eax
sub $1, %eax
cltd
idiv %eax
```

a. 0
b. 1
c. -1
d. No puede saberse con los datos del enunciado

---

9. La práctica "popcount" debía calcular la suma de bits de los elementos de un array. Un estudiante entrega lo siguiente:

```
int popcount4(unsigned* array,
                      int len) {
  int i, j, res = 0;
  for(i = 0; i < len; ++i) {
    unsigned x = array[i];
    int n = 0;
    do {
      n += x & 0x01010101L;
      x >>= 1;
    } while(x);
    for(j = 16; j == 1; j /= 2){
      n ^= (n >>= j);
    }
    res += n & 0xff;
  }
  return res;
}
```

Esta función popcount4:

a. produce el resultado correcto
b. fallaría con **`array={0,1,2,3}`**
c. fallaría con **`array={1,2,4,8}`**
d. no es correcta pero el error no se manifiesta en los ejemplos propuestos, o se manifiesta en ambos

---

10. La práctica "paridad" debía calcular la suma de paridades impar (XOR de todos los bits) de los elementos de un array. Un estudiante entrega la siguiente versión de parity6:

```
int parity6(unsigned * array,
                      int len) {
  int  i, result = 0;
  unsigned x;
  for (i=0; i<len; i++){
    x = array[i];
    asm("mov %[x], %%edx  \n\t"
        "shr $16,  %%edx  \n\t"
        "shr $8,   %%edx  \n\t"
        "xor %%edx,%%edx  \n\t"
        "setp %%dl        \n\t"
        "movzx %%dl, %[x] \n\t"
        : [x] "+r" (x)
        :
        : "edx"
        );
    result += x;
  }
  return result;
}
```

Esta función parity6:

a. produce el resultado correcto
b. fallaría con **`array={0,1,2,3}`**
c. fallaría con **`array={1,2,4,8}`**
d. no es correcta pero el error no se manifiesta en los ejemplos propuestos, o se manifiesta en ambos

---

11. En la práctica "paridad" se pide calcular la suma de paridades de una lista de enteros sin signo. Suponer que un estudiante entrega la siguiente versión:

```
int paridad5(unsigned* array,
             int len) {
  int i, k, result = 0;
  unsigned x;
  for (i = 0; i < len; i++) {
    x = array[i];
    for (k = 16; k == 1; k /= 2)
      x ^= x >> k;
    result += (x & 0x01);
  }
  return result;
}
```

Esta función:

a. es correcta

b. falla para `array={0,1,2,3}`

c. falla para `array={1,2,3,4}`

d. no se puede marcar una y sólo una de las opciones anteriores

---

12. Utilizando la sentencia asm, las denominadas restricciones que se indican al final de dicha sentencia, involucran a:

a. solamente las entradas

b. solamente las salidas

c. solamente los sobrescritos

d. Ninguna de las anteriores es cierta

---

13. En la realización de la práctica de la bomba digital, una parte del código máquina es el siguiente:

```
0x080486e8 <main+120>: call 0x8048524 <strncmp>
0x080486ed <main+125>: test %eax,%eax
0x080486ef <main+127>:  je   0x80486f6<main+134>
0x080486f1 <main+129>: call 0x8048604  <boom>
```

¿Cuál de los siguientes comandos cambiaría el salto condicional por un salto incondicional?

a. set $0x080486ef=0xeb

b. set *(char*)0x080486ef=0xeb

c. set *(char*)0x080486f6=jmp

d. set %0x080486ef=0xeb

---

14. En una bomba como las estudiadas en prácticas, del tipo…

```
0x0804873f <main+207>:  call     0x8048504 <scanf>
0x08048744 <main+212>: mov    0x24(%esp),%edx
0x08048748 <main+216>: mov    0x804a044,%eax
0x0804874d <main+221>: cmp    %eax,%edx
0x0804874f <main+223>:  je  0x8048756<main+230>
0x08048751 <main+225>: call   0x8048604 <boom>
0x08048756 <main+230>:  ...
```

la contraseña es...

a. el entero 0x804a044

b. el entero almacenado a partir de la posición de memoria 0x804a044

c. el string almacenado a partir de la posición de memoria 0x24(%esp)

d. ninguna de las anteriores

---

15. En una bomba como las estudiadas en prácticas, del tipo…

```
0x080486e8 <main+120>: call 0x8048524 <strncmp>
0x080486ed <main+125>: test   %eax,%eax
0x080486ef <main+127>:  je    0x80486f6 <main+134>
0x080486f1 <main+129>:  call  0x8048604 <boom>
0x080486f6 <main+134>:  ...
```

la contraseña es...

a. el valor que tenga %eax

b. el string almacenado a partir de donde apunta %eax

c. el entero almacenado a partir de donde apunta %eax

d. ninguna de las anteriores

---

16. El servidor de SWAD tiene dos procesadores Xeon E5540 con 4 núcleos cada uno. Cada procesador tiene 4 caches L1 de instrucciones de 32 KB, 4 caches L1 de datos de 32 KB, 4 caches unificadas L2 de 256 KB y una cache unificada L3 de 8MB. Suponga que un proceso swad, que se ejecuta en un núcleo, tiene que ordenar un vector de estudiantes accediendo repetidamente a sus elementos. Cada elemento es una estructura de datos de un estudiante y tiene un tamaño de 4KB. Si representamos en una gráfica las prestaciones en función del número de estudiantes a ordenar, ¿para qué límites teóricos en el número de estudiantes se observarán saltos en las prestaciones debidos a accesos a la jerarquía de memoria?

a. 4 / 32 / 512 estudiantes

b. 8 / 64 / 2048 estudiantes

c. 16 / 32 / 64 estudiantes

d. 32 / 256 / 8192 estudiantes

---

17. En la práctica de la cache, el código de line.cc incluye la sentencia

```
for (unsigned line=1;line<=MAXLINE;
                line<<=1) { … }
```

¿Qué objetivo tiene la expresión line<<=1?

a. Salir del bucle si el tamaño de línea se volviera menor o igual que 1 para algún elemento del vector
b. Duplicar el tamaño del salto en los accesos al vector respecto a la iteración anterior
c. Volver al principio del vector cuando el índice exceda la longitud del vector
d. Sacar un uno (1) por el stream line

---

18. Sea un computador de 32 bits con una memoria caché L1 para datos de 32 KB y líneas de 64 bytes asociativa por conjuntos de 2 vías. Dado el siguiente fragmento de código:

```
int v[262144];
for (i = 0; i < 262144; i += 2)
   v[i] = 9;
```

¿Cuál será la tasa de fallos aproximada que se obtiene en la primera ejecución del bucle anterior?

a. 0 (ningún fallo)
b. 1/2 (mitad aciertos, mitad fallos)
c. 1/8 (un fallo por cada 8 accesos)
d. 1 (todo son fallos)

---

19. Abajo se ofrece el listado de una función para multiplicar matrices C = A x B.

```
void mult_matr(   float A[N][N],
   float B[N][N], float C[N][N]){
   /* Se asume valor inicial C = {0,0...} */
int i,j,k;
for (i=0; i<N; i++)
 for (j=0; j<N; j++)
  for (k=0; k<N; k++)
   C[i][j] += A[i][k] * B[k][j];
}
```

Suponer que:

• El computador tiene una cache de datos de 8 MB, 16-vías, líneas de 64 bytes.
• N es grande, una fila o columna no cabe completa en cache.
• El tamaño de los tipos de datos es como en IA32.
• El compilador optimiza el acceso a C[i][j] en un registro.

Aproximadamente, ¿qué tasa de fallos se podría esperar de esta función para valores grandes de N?

a. 1/16
b. 1/8
c. 1/4
d. 1/2

---

20. Con los mismos supuestos, imaginar que se modifica la última sentencia (el cuerpo anidado) por esta otra

```
C[i][j] += A[i][k] * B[j][k];
```

de manera que se calcule C = A x B' (A por traspuesta de B). Aproximadamente, ¿qué tasa de fallos se podría esperar de esta nueva función para valores grandes de N?

a. 1/16
b. 1/8
c. 1/4
d. 1/2
