2º Grado Informática
Estructura de Computadores
2 Febrero 2015

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. ¿De qué tipo son los procesadores Intel que usamos en los laboratorios?

a. little-endian
b. big-endian
c. puede ajustarse mediante un bit de control en el registro CR0 que funcionen como little- o big-endian
d. el concepto de endian no es aplicable a estas máquinas, ya que un registro del procesador no cabe en una posición de memoria

2. ¿En qué registro pasa el primer argumento a una función según el estándar _cdecl en una arquitectura IA32?

a. edi
b. esi
c. eax
d. Las anteriores respuestas son erróneas

3. ¿Cuál de las siguientes instrucciones es incorrecta en ensamblador GNU/as IA32?

a. pop %eip
b. pop %ebp
c. mov (%esp),%ebp
d. lea 0x10(%esp),%ebp

4. 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

5. Si EBX=0x00002a00 y EDI=0x0000000a, ¿cuál es la dirección efectiva en la instrucción add 0x64(%ebx,%edi,2),%eax?

a. 0x00002a70
b. 0x00002a78
c. 0x00005478
d. 0x000054dc

6. Las instrucciones que asignan la dirección 0x78e00 a un puntero situado en la posición 0xff00 son:

a. mov 0x78e00,%eax; mov %eax,0xff00
b. mov 0x78e00,%eax; lea %eax,0xff00
c. mov $0x78e00,%eax; mov %eax,0xff00
d. mov $0x78e00,%eax; lea %eax,0xff00

7. Si ECX vale 0, la instrucción adc $-1,%ecx

a. Pone CF=1
b. Pone CF=0
c. Cambia CF
d. No cambia CF

8. Si el contenido de ESP es 0xAC00 antes de ejecutar la instrucción push %ebx. ¿Cuál será su contenido tras ejecutarla?

a. 0xABFC

b. 0xABFE
c. 0xAC02
d. 0xAC04

---

9. En IA32, tras dividir 0x640000 (64 bits) entre 0x8000 (32 bits), el resultado será:

a. 0x0 en AX y 0xC8 en DX.
b. 0x0 en EAX y 0xC8 en EDX.
c. 0xC8 en AX y 0xC8 en DX.
d. 0xC8 en EAX y 0x0 en EDX.

---

10. Si los contenidos de ESP y EBP son, respectivamente, 0x0008d040 y 0x00000000 antes de ejecutar una instrucción call, tras ejecutar el ret correspondiente, los contenidos serán:

a. 0x0008d040 y 0x00000000
b. 0x0008d040 y 0x0008d040
c. 0x0008d03c y 0x0008d03c
d. 0x0008d044 y 0x00000000

---

11. La instrucción leave hace exactamente lo mismo que

a. mov %ebp, %esp; pop %ebp
b. pop %eip
c. mov %esp, %ebp; pop %esp
d. ret

---

12. Sean un int*a y un int n. Si el valor de %ecx es a y el valor de %edx es n, ¿cuál de los siguientes fragmentos de ensamblador se corresponde mejor con la sentencia C return a[n]?

a. ret (%ecx,%edx,4)
b. leal (%ecx,%edx,4),%eax; ret
c. mov (%ecx,%edx,4),%eax; ret
d. mov (%ecx,%edx,1),%eax; ret

---

13. ¿Cuál de los siguientes fragmentos es correcto para comenzar un programa en ensamblador que conste de un solo archivo .s?

a.
```
.text
_start:
```
b.
```
.section .text
.global _start
_start:
```
c.
```
.section .text
.local _start
_start:
```
d.
```
section .text
.start _global
_start:
```

---

14. En la práctica 2 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. **0x07ff ffff**
b. **0x0800 0000**
c. **0xfbff ffff**
d. **0xfc00 0000**

---

15. 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"
        "setpe %%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. No es correcta, fallaría por ejemplo con **array={0,1,2,3}**
c. No es correcta, fallaría por ejemplo 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

---

16. Suponga la siguiente sentencia asm en un programa:

asm(“ add (%[a],%[i],4),%r”

:[r] “+r” (result)

:[i] “r” (i),

[a] “r” (array));

¿Cuál de las siguientes afirmaciones es incorrecta?

a. r es un registro de entrada/salida
b. a es un registro de entrada
c. i es un registro de salida
d. se desea que el valor calculado por la instrucción ensamblador quede almacenado en la variable C result

---

17. 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

---

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

```
for (unsigned long long line=1;
line<=LINE; 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

---

19. En la práctica de la cache, el código de size.cc accede al vector saltando de 64 en 64. ¿Por qué?

a. Porque cada elemento del vector ocupa 64 bytes
b. Para recorrer el vector más rápidamente
c. Porque el tamaño de cache L1 de todos los procesadores actuales es de 64KB
d. Para anular los aciertos por localidad espacial, esto es, que sólo pueda haber aciertos por localidad temporal

---

20. En el programa size de la práctica de la cache, si el primer escalón pasa de tiempo=2 para un tamaño de vector menor que 32KB a tiempo=8 para un tamaño mayor que 32KB, podemos asegurar que:

a. La cache L1 es cuatro veces más rápida que la cache L2.
b. La cache L1 es seis veces más rápida que la cache L2.
c. La cache L1 es al menos cuatro veces más de rápida que la cache L2.
d. La cache L1 es como mucho cuatro veces más rápida que la cache L2.
