Teste 1A de Arquitetura de Computadores

17/4/2019 Duração 1h45 *sem consulta*

As perguntas com múltiplas respostas têm de ser respondidas na folha de respostas anexa.
Respostas erradas descontam 1/5 da cotação da pergunta ou alínea.

Número: _____ *Nome:* _____

1 (2 val) Considere uma representação de inteiros usando 7 bits.

a) Como é codificado o valor 25 (sem sinal)?

A) 001 1001 B) 010 0101 C) 101 0101 D) 100 0101 E) 110 0101

b) Como é codificado –1 (usando a norma complemento para 2)?

A) 100 0001 B) 100 0000 C) 001 0001 D) 111 1111 E) 000 0001

c) Qual é o maior valor que se pode representar (sem sinal, resposta em base 10) ?

A) 128 B) 256 C) 511 D) 127 E) 255

d) Qual é o menor valor que se pode representar usando complemento para 2 (resposta em base 10) ?

A) –1 B) –128 C) –64 D) –127 E) –63

e) Qual é o menor valor que se pode representar usando complemento para 2 (resposta em base 2) ?

A) 111 1111 B) 011 1111 C) 100 0000 D) 100 1111 E) 000 1111

2 (0,6 val) No contexto da representação de valores inteiros com sinal, usando a norma complemento para 2, qual é a representação dos valores 7 e -7, utilizando 8 e 10 bits?

A) 7 (8 bits): 00000111 7 (10 bits): 0000000111 -7 (8 bits): 11111001 -7 (10 bits): 1111111001

B) 7 (8 bits): 00000111 7 (10 bits): 0000011100 -7 (8 bits): 11111001 -7 (10 bits): 1111111001

C) 7 (8 bits): 00000111 7 (10 bits): 0000000111 -7 (8 bits): 11111001 -7 (10 bits): 0011111001

D) 7 (8 bits): 00000111 7 (10 bits): 0000011100 -7 (8 bits): 10000111 -7 (10 bits): 1000000111

E) 7 (8 bits): 00000111 7 (10 bits): 0000000111 -7 (8 bits): 10000111 -7 (10 bits): 1000000111

3 (1,5 val) Considere operações sobre inteiros de 6 bits com sinal, representados na norma *complemento para dois*. A unidade aritmética suporta as seguintes flags: ZF (*zero flag*), CF (*carry flag*), OF (*overflow flag*), SF (*sign flag*). Indique qual o valor correto das flags para as seguintes operações:

a) 17 - 17

A) ZF=1, CF=0, OF=0, SF=0 B) ZF=1, CF=1, OF=0, SF=0 C) ZF=1, CF=1, OF=1, SF=0
D) ZF=0, CF=1, OF=0, SF=0 E) ZF=0, CF=0, OF=0, SF=0

b) 17 + 17

A) ZF=0, CF=0, OF=0, SF=1 B) ZF=0, CF=1, OF=0, SF=0 C) ZF=0, CF=1, OF=1, SF=0
D) ZF=0, CF=1, OF=0, SF=1 E) ZF=0, CF=0, OF=1, SF=1

c) -17 + 20

A) ZF=0, CF=0, OF=0, SF=1 B) ZF=0, CF=1, OF=0, SF=0 C) ZF=0, CF=1, OF=1, SF=0
D) ZF=0, CF=1, OF=0, SF=1 E) ZF=0, CF=0, OF=1, SF=1

4 (0,8 val) Qual das seguintes expressões coloca a 1 os **bits 4 e 2** do valor guardado na variável x, sem alterar os demais bits?
(note: o bit zero é o menos significativo)

A) x = x | 4 | 2; B) x = x & 4 & 2; C) x = x | 20;

D) x = x & 20; E) x = x << 4 << 2;

5 (2,1val) No contexto da representação binária de números com vírgula flutuante:
a) Qual é a representação binária do número 0.1875?

A) 0.11 B) 1.1 C) 0.011 D) 0.0011 E) 0.0001

b) Qual é a representação do número 0.1875 na norma IEEE-754 de precisão simples?
(recorde que nesta norma usa-se 1 bit sinal, 8 bits para expoente mais 127 e 23 bits para a parte fracionária da mantissa)

A) `00000001110000000000000000000000`

B) `00111110010000000000000000000000`

C) `01000000111100000000000000000000`

D) `01000000010000000000000000000000`

E) `00111110011000000000000000000000`

c) Qual é o valor em base 10 do expoente representado no padrão de bits `01011001010000000000000000000000` da norma IEEE-754 de precisão simples?

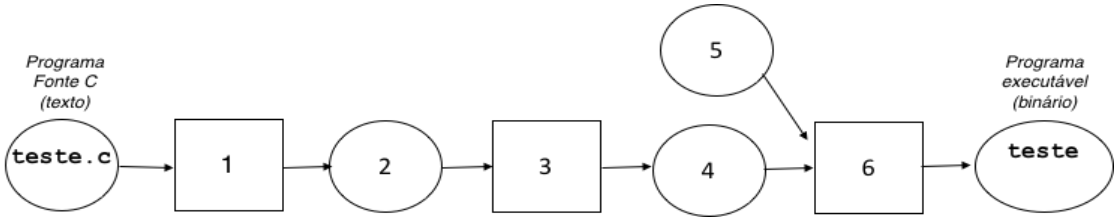
A) 2473901162496000 B) 178 C) 1.5 D) -78 E) 51

6 (0,6 val) Qual é o resultado da aplicação da seguinte função f ao número 19? `f(19)`

```
unsigned int f(unsigned int n) {
    int count = 0;
    while (n != 0) {
        count += n & 1;
        n = n >> 1;
    }
    return count;
}
```

A) 19 B) 3 C) 2 D) 1 E) 0

7 (1 val)
a) A figura abaixo representa um processo de compilação de um programa fonte em C com o comando: `cc -g -o teste teste.c`. O processo tem três passos (representados pelos quadrados 1, 3 e 6) e faz uso de três ficheiros (representados pelas circunferências 2, 4 e 5). Os ficheiros 2 e 4 representam estágios intermédios do processo e normalmente são temporários. Selecione a opção que apresenta a legenda correta para o processo, nomeadamente para os passos e ficheiros numerados de 1 a 6.



A) [1] Assembler (as) [2] Ficheiro objeto (.o) [3] Compilador
[4] Debugger [5] Ficheiro das classes [6] Ligador/Linker (ld)

B) [1] Compilador [2] Ficheiro objeto (.o) [3] Ligador/Linker (ld)
[4] Ficheiro executável [5] terminal [6] debugger

C) [1] Compilador [2] Ficheiro assembly [3] Assembler
[4] Ficheiro objeto (.o) [5] Bibliotecas [6] Ligador/Linker (ld)

D) [1] Editor de texto [2] Compilador [3] Ficheiro objeto (.o)
[4] Ligador/Linker (ld) [5] Bibliotecas [6] Debugger

E) [1] Compilador (javac) [2] Ficheiro com código JVM [3] Ligador/Linker
[4] Ficheiro binário .class [5] Bibliotecas de classes [6] Interpretador (JVM)

b) Indique as linhas de comandos necessárias para "*assemblar*" e obter o executável "teste" para um programa escrito em *assembly* no ficheiro "teste.s".

A) as -o teste teste.s
ld -g teste

B) as -g teste teste.s
ld -o teste.s

C) as -o teste.o teste.s
ld -o teste teste.o

D) ld -o teste teste.s
as teste.o

E) as -g teste.o teste.s;
ld -o teste.o

8 (0,8 val) A arquitetura de Von Neumann caracteriza-se principalmente por (indique a frase verdadeira):

A) O programa está codificado na memória, tal como os dados, sendo interpretado pelo CPU

B) O programa está codificado num periférico de onde o CPU vai lendo e interpretando cada instrução

C) Os dados estão codificados em grupos de 8 bits (bytes)

D) Os dados estão codificados em grupos de 8 dígitos decimais

E) Esta arquitectura é realizada por circuitos lógicos electromecânicos

9 (0,8 val) A arquitetura interna de um processador (CPU) inclui como principais componentes (escolha a opção verdadeira):

A) Memória, registos e ALU (unidade aritmética e lógica)

B) Memória, bus de sistema e registos

C) Registos, unidade de controlo e ALU (unidade aritmética e lógica)

D) Memória, bus de sistema, registos e ALU (unidade aritmética e lógica)

E) Registos, unidade de controlo, ALU (unidade aritmética e lógica) e unidades periféricas

10 (0,8 val) O processador (CPU) interpreta sequencialmente as instruções máquina de um programa executando repetidamente um ciclo. Escolha a opção que apresenta o processamento realizado neste ciclo pela ordem correta:

A) *fetch* de uma instrução em memória; obter os operandos; descodificação da instrução; execução da instrução

B) *fetch* de uma instrução num periférico; descodificação da instrução; obter os operandos; execução da instrução

C) *fetch* de uma instrução em memória; descodificação da instrução; obter os operandos; execução da instrução

D) *fetch* de uma instrução num periférico; descodificação da instrução; execução a instrução; obter os operandos

E) obter os operandos; *fetch* da instrução em memória; descodificação da instrução; executar a instrução

11 (0,8 val) Uma determinada arquitectura suporta instruções máquina com variados tipos de operandos. Estes podem estar imediatamente na instrução, num registo ou em memória. Indique qual o tipo de operando que pode tornar mais lenta a execução da instrução.

A) O operando imediatamente na instrução será mais lento porque é necessário descodificar primeiro a instrução.

B) O operando num registo será mais lento porque é necessário transferir o seu conteúdo para memória para se poder executar a instrução.

C) O operando num registo será mais lento porque é muito lento transferir o seu conteúdo para a ALU.

D) O operando em memória será mais lento porque é muito lento transferir o seu conteúdo para a ALU.

E) O operando imediatamente na instrução será mais lento porque é necessário transferir este para memória para se poder executar a instrução.

12 (0,8 val) Considere a seguinte sequência de instruções em assembly IA32/Linux. Quais os valores nos registos EAX e EBX após a sua execução?

```
movl $5, %eax
movl $3, %ebx
addw %bx, %ax
inc %bx
```

A) EAX = 5, EBX = 3 B) EAX = 8, EBX = 3 C) EAX = 5, EBX = 8 D) EAX = 8, EBX = 8 E) EAX = 8, EBX = 4

13 (0,8 val) Considere a seguinte sequência de instruções em assembly IA32/Linux que efectua um ciclo. Quantas vezes o ciclo executa e qual o valor final no registo EBX?

```
        movl $4, %eax
        movl $0, %ebx
ciclo:  add %eax, %ebx
        dec %eax
        jnz ciclo
```

A) 4 vezes, EBX = 4 B) 3 vezes, EBX = 10 C) 5 vezes, EBX = 4 D) 4 vezes, EBX = 10 E) 5 vezes, EBX = 10

14 (0,8 val) Considere o programa em C apresentado em baixo. Assumindo que as variáveis ficam seguidas em memória pela ordem que foram declaradas, qual é o valor correto para os vetores 'v1' e 'v2' no final da execução do programa (imediatamente antes do return final)?

```
int main () {
  double v1[4] = {1.0, 1.5, 2.0, 2.5};
  double v2[2] = {5.0, 5.5};
  double *p1 = &v1[2];
  double a = *v1;
  *v2 = 3.14;
  p1[1] = a;
  return 0;
}
```

A) v1={1.0, 1.5, 2.0, 2.5}; v2={1.0, 5.5};

B) v1={1.0, 1.5, 2.0, 1.0}; v2={3.14, 5.5};

C) v1={1.0, 1.5, 2.0, 2.5}; v2={1.0, 3.14};

D) v1={3.14, 1.5, 1.0, 2.5}; v2={3.14, 5.5};

E) v1={1.0, 1.5, 1.0, 2.5}; v2={5.0, 5.5};

15 (0,8 val) Considere o programa em C apresentado em baixo. Escolha a afirmação VERDADEIRA!

```
1 #include <stdio.h>
2
3 int main () {
4   char *s = "hello!";
5   foreach (char c: s) {
6     printf ("%d,", c);
7   }
8   return 0;
9 }
```

A) O programa compila sem erros, executa com sucesso e tem como output "h,e,l,l,o,!,"

B) O programa compila sem erros , executa com sucesso e tem como output "104,101,108,108,111,33"

C) O programa compila com um warning na linha 6, executa e tem como output "h,e,l,l,o,!,"

D) O programa compila com um warning na linha 6, executa com sucesso e tem como output "104,101,108,108,111,33"

E) O programa não compila, com um erro sintático na linha 5.

16 (0,8 val) Quando utilizamos a linguagem de programação C, o que acontece quando atribuímos um valor do tipo double a uma variável do tipo int, como acontece na instrução "`int x = 3.95`"?

A) O compilador não compila e dá um erro de compilação.

B) O compilador compila e durante a execução o valor atribuído a 'x' é 3.

C) O compilador compila e durante a execução o valor atribuído a 'x' é 4.

D) O compilador compila e durante a execução o valor atribuído a 'x' é 3.95.

E) O compilador compila, mas quando se executa o programa este termina nesta instrução de atribuição com um erro de *segmentation fault*.

17 (1 val) Considere uma arquitetura *little-endian* na qual os registos de dados têm 16 bits e os de endereço 24 bits. Num programa em C declarou as seguintes variáveis:

```
int a;
int *b;
char c[6] = "ABCD\n";
char *d;
```

Sabendo que estas estão sequencialmente em memória, respetivamente nos endereços 100, 102, 105 e 111, indique qual das seguintes figuras é uma representação válida da memória do computador (cada célula representa um byte), após executar as seguintes instruções "de inicialização":

```
a = 17;
b = &a;
d = &c;
```

A)

End.	Cont.
100	0
101	17
102	0
103	0
104	100
105	0
106	\n
107	D
108	C
109	B
110	A
111	0
112	0
113	105
114	0
115	0

B)

End.	Cont.
100	17
101	0
102	100
103	0
104	0
105	0
106	A
107	B
108	C
109	D
110	\n
111	0
112	105
113	0
114	0
115	0

C)

End.	Cont.
100	17
101	0
102	100
103	0
104	0
105	A
106	B
107	C
108	D
109	\n
110	0
111	105
112	0
113	0
114	0
115	0

D)

End.	Cont.
100	17
101	0
102	100
103	0
104	0
105	0
106	A
107	B
108	C
109	D
110	\n
111	105
112	0
113	0
114	0
115	0

E)

End.	Cont.
100	17
101	0
102	100
103	0
104	0
105	0
106	0
107	\n
108	D
109	C
110	B
111	A
112	105
113	0
114	0
115	0

18 (0,8 val) Considere o seguinte programa em C. Identifique qual é o output correto.

```
#include <stdio.h>

int main () {
  char *s = "abcdefg";
  int a = strlen(s);
  s[3] = '\0';
  int b = strlen(s);
  s[5] = '\0';
  int b = strlen(s);
  printf ("a=%d b=%d c=%d\n", a, b, c);
  return 0;
}
```

A) a=7, b=3, c=3
B) a=7, b=3, c=5
C) a=7, b=7, c=7
D) a=6, b=2, c=4
E) a=6, b=2, c=2

19 (0,8 val) Considere o seguinte programa em C. Identifique qual é o output correto.

```
#include <stdio.h>

int main () {
  float f1 = 3.7587;
   int i1 = (int)f1;
   float f2 = (float)i1;
   int i2 = (int)f2;
   printf ("f1=%f, i1=%d, f2=%f, i2=%d\n", f1, i1, f2, i2);
   return 0;
}
```

A) f1=3.758700, i1=4, f2=4.000000, i2=4

B) f1=3.758700, i1=3, f2=3.000000, i2=3

C) f1=3.758700, i1=4, f2=3.000000, i2=3

D) f1=3.758700, i1=3, f2=3.758700, i2=3

E) f1=3.758700, i1=4, f2=4.000000, i2=3

20 (1,5 val)

a) Num programa em C estamos a representar algarismos pelo carácter respectivo em vez de usar inteiros. Implemente uma função que permita efetuar a subtração entre algarismos representados por `char`. Se a subtração não for possível (quando daria um resultado negativo) devolve o carácter '?'. Veja os exemplos de utilização:

```
char c = subChar('9', '8');  // c fica com '1'
char d = subChar('8', '9');  // d fica com '?'
```

Complete a implementação da função:

```
char subChar( char a1, char a2 ){

    if (_____a1-a2<0____) return ____'?'_____________;


    else return _____a1-a2+'0'_____________;

}
```

b) Recorde que dividir um número real por dois é equivalente a multiplicar por 2^{-1}. Considerando a representação em vírgula flutuante usada nos computadores (IEEE 754), tal operação pode ser obtida subtraindo 1 à representação do expoente. Complete o código seguinte para efetuar a divisão indicada:

```
float div2(float f) {
  int x = *(int*)&f;

  int e = (___x>>23___) & 0xFF; // obter representacao do expoente

  e = ____e-1_________;

  int res = x & (_____0x807FFFFF_______);

  res = res | (______e<<23__________);

  return *(float*)&res;
}
```