MINISTÉRIO DA EDUCAÇÃO

INSTITUTO FEDERAL DO ESPÍRITO SANTO

REITORIA

Avenida Rio Branco, 50 – Santa Lúcia – 29056-255 – Vitória – ES

27 3357-7500

CONCURSO PÚBLICO
EDITAL Nº 03 / 2014

Professor do Magistério do Ensino Básico, Técnico e Tecnológico

ÍNDICE DE INSCRIÇÃO	311
CAMPUS	Serra
ÁREA/SUBÁREA/ESPECIALIDADE	CIÊNCIAS DA COMPUTAÇÃO - SISTEMAS DA COMPUTAÇÃO

PROVA DE CONHECIMENTOS ESPECÍFICOS | DISCURSIVA
MATRIZ DE CORREÇÃO

QUESTÃO 01

Opção em liguagem C:

a)

```
// mat é a matriz de entrada; lin é quantidade de linhas da matriz de entrada e col é a quantidade de
// colunas da matriz de entrada

int** seta2 (int lin, int col, int *mat)
{
  int i, j;
  int ** trp;

  trp = (int **)malloc(col*sizeof(int*));

  for (i = 0; i < col; i++)
    trp[i] = (int *) malloc (lin*sizeof(int));

  for (i = 0; i < lin; i++)
    for (j = 0; j < col; j++)
      trp[j][i] = mat[i*col+j];

  return trp;
}
```

b)

```
// mat é a matriz de entrada; lin é quantidade de linhas da matriz de entrada e col é a quantidade de
// colunas da matriz de entrada

int* seta3 (int lin, int col, int **mat)
```

```
{
  int i, j;
  int * inv;

  inv = (int *)malloc(lin*col*sizeof(int));

  for (i = (lin - 1); i >= 0; i--)
    for (j = (col - 1); j >= 0; j--)
      inv[(lin*col-1)-(j*lin+i)] = mat[i][j];

  return inv;
}
```

Opção em linguagem Pascal:

```
type

  tpArrayChar = array of char;
  tpArrayArrayChar = array of tpArrayChar;

a)
function seta2 (lin:integer; col:integer; mat: tpArrayChar): tpArrayArrayChar;
var
  i, j : integer;
  trp: tpArrayArrayChar;

begin
  setlength(trp, col);
  for i := 0 to (col -1 ) do
    setlength(trp[i], lin);

  for i := 0 to (lin - 1) do
    for j := 0 to (col - 1) do
      trp[j, i] := mat[i*col+j];

  seta2 := trp;
end;

b)
function seta3 (lin:integer; col:integer; mat: tpArrayArrayChar): tpArrayChar;
var
  i, j:integer;
  inv: tpArrayChar;

begin
  setlength(inv, lin*col);

  for i := (lin - 1) downto 0 do
    for j := (col - 1) downto 0 do
      inv[(lin*col-1)-(j*lin+i)] := mat[i,j];

  seta3 := inv;
end;
```

Observações:

1) Caso o candidato tenha preferido manipular os caracteres como o tipo 'char' ao invés do tipo 'int' na opção em linguagem C, sua resposta será aceita, desde que a manipulação esteja correta.

2) A questão é dividida em 2 itens, a e b, onde cada item corresponde a 50% do valor da questão. Para cada item, a e b, a solução pode ser dividida em 3 partes, de igual pontuação:

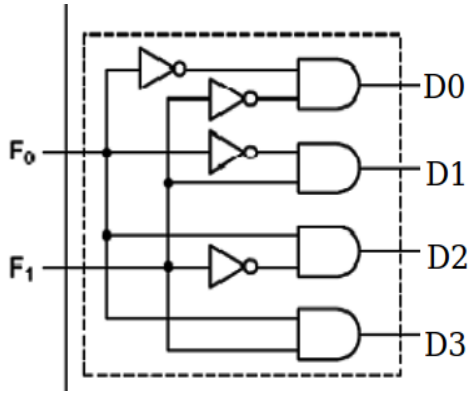
- cabeçalho da função

- alocação dinâmica do resultado da função
- preenchimento do resultado da função

QUESTÃO 02

a) [equivale a 45% da pontuação da questão]

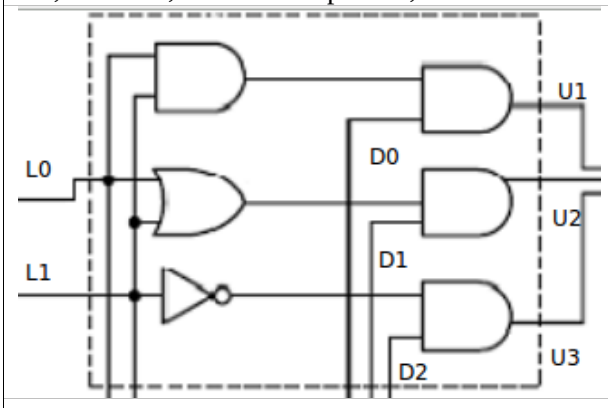
O circuito no retângulo tracejado posicionado na porção inferior esquerdo com as entradas F_0 e F_1 (apresentado abaixo) é um decodificador 2 para 4 (2x4). Um decodificador é um circuito combinacional que converte um código binário de N bits de entrada em M linhas de saída (em que N pode ser qualquer inteiro e M é um inteiro menor ou igual a 2^N), de modo que cada linha de saída será ativada por uma única combinação das possíveis de entrada.



Segue a tabela verdade do circuito:

F_0	F_1	D0	D1	D2	D3
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1

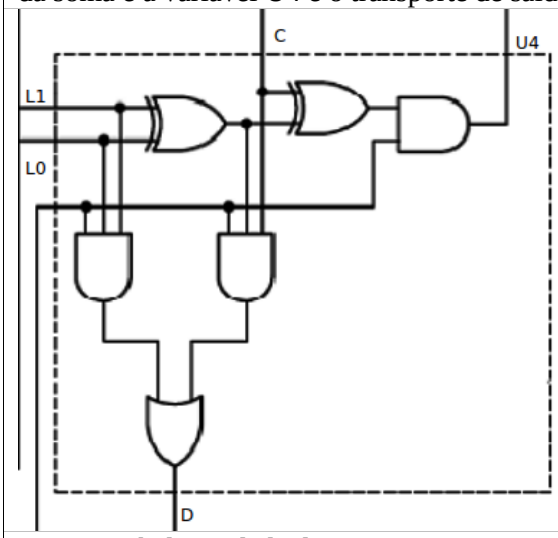
O circuito no retângulo tracejado posicionado na parte superior (apresentado abaixo) é a parte da ULA simples de 1 bit que trata das operações lógicas: AND, OR e NOT. L0 e L1 são os operandos, a saída será nas variáveis U1, U2 e U3, controlados por D0, D1 e D2.



Segue a tabela verdade do circuito:

D0	D1	D2	U1	U2	U3
1	0	0	L0 AND L1	0	0
0	1	0	0	L0 OR L1	0
0	0	1	0	0	NOT L1

O circuito no retângulo tracejado posicionado na parte inferior direita (apresentado abaixo) é um somador completo. Os operandos são as entradas L0 e L1, e o transporte de entrada (carry in) é a entrada C; o resultado da soma é a variável U4 e o transporte de saída (carry out) é a variável D.



Segue a tabela verdade do circuito:

L0	L1	C	U4 = L0+L1+C	D
0	**0**	**0**	**0**	**0**
0	**0**	**1**	**1**	**0**
0	**1**	**0**	**1**	**0**
0	**1**	**1**	**0**	**1**
1	**0**	**0**	**1**	**0**
1	**0**	**1**	**0**	**1**
1	**1**	**0**	**0**	**1**
1	**1**	**1**	**1**	**1**

Como resposta, aceita-se apenas a descrição do circuito ou a tabela verdade ou mesmo os mintermos dos circuitos.

b) [equivale a 10% da pontuação da questão]
As entradas ENA e ENB são entradas de ENABLE, isto é, servem para controlar a operação de um circuito. O pino de ENABLE permite (habilita) ou impede (desabilita) a operação do circuito. No caso do circuito desta questão, quando ENA estiver em 1 habilita a entrada A, quando ENA estiver em 0, desabilita a entrada A, pois a saída do operador lógico AND sempre será 0. O mesmo raciocínio é aplicado à ENB com relação à entrada B.

c) [equivale a 45% da pontuação da questão]

F0	F1	ENA	ENB	Output	D
0	**0**	**1**	**1**	**A AND B**	**X**
0	**1**	**1**	**1**	**A OR B**	**X**
1	**0**	**1**	**1**	**NOT B**	**X**
1	**1**	**1**	**1**	**A + B + C**	**CARRY OUT**
0	**0**	**0**	**1**	**0**	**X**
0	**1**	**0**	**1**	**B**	**X**
1	**0**	**0**	**1**	**NOT B**	**X**
1	**1**	**0**	**1**	**B + C**	**CARRY OUT**
0	**0**	**1**	**0**	**0**	**X**
0	**1**	**1**	**0**	**A**	**X**

1	**0**	**1**	**0**	**1**	**X**
1	**1**	**1**	**0**	**A + C**	**CARRY OUT**
0	**0**	**0**	**0**	**0**	**X**
0	**1**	**0**	**0**	**0**	**X**
1	**0**	**0**	**0**	**1**	**X**
1	**1**	**0**	**0**	**C**	**0**

QUESTÃO 03

a)

(1) Cite os modos de detecção do canal físico e lógico [0,6 pto]. (2) Explique que na detecção física, o transmissor monitora o canal para verificar sua utilização, se estiver ocupado aguarda, caso contrário transmite [0,6 pto]. (3)Explique que, em caso de colisão, a estação transmissora escolhe aleatoriamente um intervalo de tempo antes de tentar retransmitir eque o conjunto de valores, dos quais será sorteado o tempo de espera dobra a cada tentativa frustrada [0,6 pto].

(4)Explique que na detecção lógica, o transmissor envia um quadro RTS para o receptor que responde com um quadro CTS. As demais estações conseguem saber o tempo de duração do quadro de dados. Ao terminar de receber o quadro de dados, o receptor envia um quadro ACK. As demais estações, ao desejarem transmitir e detectarem os quadros RTS e CTS entram em estado de inatividade até o fim da transmissão, incluindo o ACK [0,6 pto].

(5) Explique que o algoritmo de backoff exponencial é usado para garantir que as estações não aumentem as colisões por tentativas repetidas de transmissão em intervalos de tempo muito curtos, pois o transmissor não pode saber quantas são as estações envolvidas na colisão e que quanto maior for intervalo de espera, desde que aleatório para cada estação, menor o risco de colisões [0,6 pto].

b)

(1) Explique que consultas recursivas são feitas respeitando-se a hierarquia da árvore DNS, desde os servidores-raiz até o servidor autoritativo responsável pelo domínio onde está registrada a máquina alvo [2 pto] .
(2)Explique que neste tipo de consulta, a resposta do servidor será sempre indicando qual é o próximo autoritativo hierarquicamente abaixo, respeitando o FQDN origalmente pesquisada e que, portanto, a resposta ao cliente será uma referência de qual servidor pode encontrar o alvo [2 pto] .

(3)Explique que a consulta iterativa retorna o IP ou uma negativa ao pedido de tradução [2 pto] .

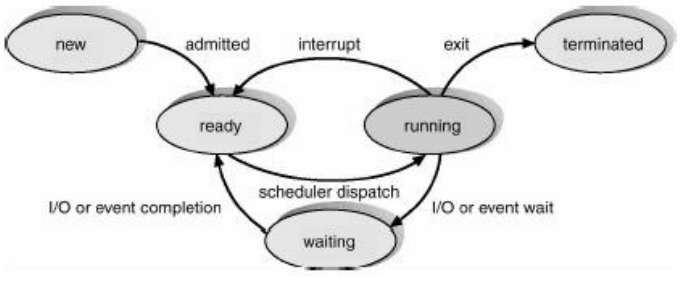
c)

Hand-shake TCP com números de sequência indicados [2 pto]
Troca de informações com incremento do Num Seq e ACK correspondentes [2 pto]
Fim da comunicação com pedido de término vindo do cliente e posterior aceito pelo servidor [2 pto]

QUESTÃO 04

a)

Diagrama [0,9 pto]



1. New/Novo – processo criado [0,3 pto] – transição automática para Pronto [0,2 pto]
2. Ready/Pronto – processo pronto para execução [0,3 pto] – transição via política de escalonamento para Execuntando [0,2 pto]
3. Running/Executando – processo em execução pela CPU [0,3 pto] – transição de volta para Pronto após

interrupção [0,2 pto] ou transição para Bloqueado depois de solicitar I/O ou evento [0,2 pto] ou transição para terminado [0,2 pto]

4. Waitin/Bloqueado – aguardando recurso ou evento [0,3 pto] – transição para Pronto após conclusão de I/O ou evento [0,2 pto]
5. Terminated/ terminado – processo finalizado [0,3 pto]

b)

1. Texto – onde reside o código de instruções e constantes; [0,9 pto]
2. Dados – variáveis declaradas no programa, inicializadas ou não;[0,9 pto]
3. Pilha – estrutura de pilha para chamadas de funções, procedimentos e respectivos retornos [0,9 pto]
4. Heap – variáveis dinâmicas criadas em tempo de execução[0,9 pto]

c)

Escalonador de processos: controla a execução dos processos no sistema operacional [0,7 pto] , executa a política de escalonamento (FIFO, LIFO, etc.) [0,8 pto], contabilizar indicadores de execução dos processos (tempo de cpu, tempo esperando I/O) para decisões futuras [0,7 pto]

Escalonador e Finalidade do S.O: os sistemas podem ser projetados para finalidades específicas (online, offline, etc.) [0,7 pto] o escalonador estabelece a política de execução de processos conforme finaloidade do sistema [0,7 pto].

d)

Sleep-Wakeup – um processo trabalha na região crítica enquanto o outro está dormindo (bloqueado). Ao terminar seu trabalho, o processo que finalizou sua tarefa acorda o outro e se coloca a dormir. [1,3 pto]

Busy Wait – um processo fica em execução mesmo que esteja esperando a liberação de um recursos utilizado por outro processo. Quando o escalonador o interrompe e o colocar na fila de pronto, outros processos poderão executar na quela mesma região crítica. [1,3 pto]

Busy Wait consome CPU sem necessidade, enquanto Sleep-Waekup coordena os processos com menor uso de CPU. [1 pto]

e)

Semáforo – é uma variável especial protegida que tem como função o controle de acesso a uma região compartilhada (região crítica) num ambiente multitarefa [1 pto].

O valor de um semáforo indica quantos processos (ou threads) podem ter acesso a um recurso compartilhado [0,5 pto].

Inicialização: Recebe um valor inteiro indicando a quantidade de processos que podem acessar um determinado recurso. [0,5 pto]

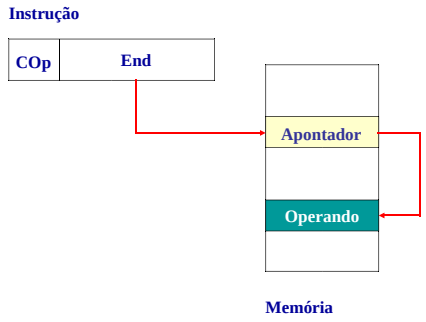
Operação wait ou P: Decrementa o valor do semáforo. Se o semáforo está com valor zero, o processo é posto para dormir.[0,5 pto]

Operação signal ou V: Se o semáforo estiver com o valor zero e existir algum processo adormecido, um processo será acordado. Caso contrário, o valor do semáforo é incrementado. [0,5 pto]

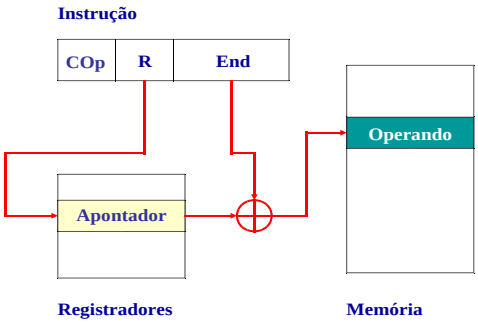
As operações de incrementar e decrementar devem ser operações atômicas, ou indivisíveis, ou seja, enquanto um processo estiver executando uma dessas duas operações, nenhum outro processo pode executar outra operação sob o mesmo semáforo, devendo esperar que o primeiro processo encerre a sua operação sob o semáforo. Essa obrigação evita condições de disputa entre vários processos. [0,6 pto]

QUESTÃO 05

a)



Modo Indireto



Modo Deslocamento (Offset)

Modo Indireto: campo de endereço referencia um apontador em memória, que referencia o operando[1,3 pto]. Duas referências à memória são feitas - mais lento [0,5 pto]

Modo Offset: O endereço efetivo é obtido através da soma de um endereço base (no registrador ou na instrução) a um deslocamento (adicional) na instrução (ou no registrador)[1,3 pto]. Uma única referência à memória é feitas e as subsequentes são utilizados valores incrementais [0,5 pto]

b)

Gargalo de Von Neumann – a memória é o elemento crítico no fluxo de processamento de instruções, uma vez que maior parte do tempo é gasto no deslocamento de dados entre a memória e as unidades da CPU [2 pto].

O tamanho da instrução deve guardar relação com o espaço de endereços da memória. O formato deve decidir quantos operandos serão recuperados da memória para a execução de uma instrução. Busca-se minimizar o número de acessos á memória,l evitando-se modos indiretos de endereçamento[1,6 ptos].

c)

Pipeline – separar as operações do ciclo busca-decodifica-executa e unidades independentes de execução [1 ptos], formando uma linha de montagem onde as instruções sejam processadas de forma contínua. Assim que as unidades inicias façam seu trabalho, uma nova instrução pode ser trazida para execução melhorando o Throughput da CPU[1 pto] .

O desempenho da CPU tende a melhorar, uma vez que múltiplas instruções estarão, em estágios diferentes, em execução na CPU [1,6 pto].

d)

Classe 1 - Controle: saltos condicionais (IF) que podem mudar a sequência das instruções a serem executadas e demandar o esvaziamento do Pipiline [1,8 pto]

Classe 2 – Dados: a dependência de certas instruções com relação à produção de dados por instruções antecessoras. [1,8 pto]

Classe 3 – Recursos: a disputa de instruções por recursos da CPU ou da memória, impedindo a realização [1,8 pto]

Obs.: Apenas 2 das três opções. Se forem explicadas as três classes, cada uma valerá 1,2 ptos