UNIP
UNIVERSIDADE PAULISTA

CIÊNCIA DA COMPUTAÇÃO

MATERIAL INSTRUCIONAL ESPECÍFICO

TOMO 8

CQA - COMISSÃO DE QUALIFICAÇÃO E AVALIAÇÃO

CIÊNCIA DA COMPUTAÇÃO

MATERIAL INSTRUCIONAL ESPECÍFICO

TOMO 8

Christiane Mazur Doi

Doutora em Engenharia Metalúrgica e de Materiais, Mestra em Ciências - Tecnologia Nuclear, Especialista em Língua Portuguesa e Literatura, Engenheira Química e Licenciada em Matemática, com Aperfeiçoamento em Estatística. Professora titular da Universidade Paulista.

Tiago Guglielmeti Correale

Doutor em Engenharia Elétrica, Mestre em Engenharia Elétrica e Engenheiro Elétrico (ênfase em Telecomunicações). Professor titular da Universidade Paulista.

Questão 1

Questão 1.[1]

O código a seguir mostra um programa escrito na linguagem de programação Java.

```
public class Java {
	private static void trocaB(String a, String b){
		String tmp = a;
		a = b;
		b = tmp;
	}
	private static void trocaC(int array[], String a){
		for(int x = 0; x < array.length; x++){
			array[x] = array[x] * Integer.valueOf(a);
		}
	}
	public static void main(String[] args){
		int array[] = {1,2,3,4,5};
		String a = "2", b = "5";
		trocaB(a, b);
		trocaC(array, a);
		System.out.print(a + " " + b + " ");
		for(int x = 0; x < array.length; x++){
			System.out.print(array[x] + " ");
		}
	}
}
```

Após ser executado o código, o valor impresso na saída padrão do usuário será

A. 5 2 5 10 15 20 25.

B. 2 5 2 4 6 8 10.

C. 5 2 2 4 6 8 10.

D. 5 2 1 2 3 4 5.

E. 2 5 1 2 3 4 5.

[1]Questão 35 – Enade 2014.

1. Introdução teórica

Teste de mesa

Um programador deve conhecer o funcionamento dos algoritmos que utiliza em seu cotidiano.

Uma das melhores formas de se compreender o funcionamento de um algoritmo é simulando, manualmente, seu funcionamento com lápis e papel. Para isso, monta-se um quadro com os valores das variáveis utilizadas em métodos e em funções (dependendo da linguagem a ser utilizada), conforme quadro 1 a seguir, que mostra o chamado modelo de teste de mesa.

Quadro 1. Modelo de teste de mesa.

Variáveis / Linha	Variável 1	Variável 2	Variável 3	Variável 4
...	...	...	...	...
10	2	44	"casa"	-
11	3	44	"casa"	1
12	4	44	"casa"	1
...	...	...	...	...

Cada linha do quadro corresponde à execução de uma linha do programa. Deve-se ter o cuidado de acompanhar cuidadosamente o ponto do programa executado em dado instante e marcá-lo, com o lápis, no código do programa. É bastante comum, também, que se reserve uma área no papel para a saída do programa. Nessa área, obtém-se o resultado da execução dos métodos que apresentam algum tipo de saída na tela, como o método "print".

Na realidade, o computador não está executando exatamente cada linha do programa em Java, por exemplo, uma de cada vez, como executado no papel. O motivo disso é que a linguagem Java exige a utilização de um compilador. Na linguagem Java, o processo de compilação gera um código intermediário chamado de *bytecode*, que é executado pela máquina virtual Java (*JVM-Java Virtual Machine*). Esse código intermediário é composto pelo conjunto de instruções da máquina virtual Java e é similar, em alguns aspectos, a um binário executado diretamente por uma máquina real.

Isso significa que a máquina virtual Java não está executando diretamente o código que escrevemos em Java, mas sim o código gerado pelo compilador, de forma que o código que está realmente sendo executado pode ser muito maior do que o programa original. Assim, uma única linha do programa Java pode corresponder a um grande número de linhas do programa executado pela máquina virtual.

2. Resolução da questão

Para compreender o programa de forma mais fácil, não é preciso numerar todas as linhas de forma consecutiva: pode-se trabalhar de forma isolada com cada um dos métodos, entender seu funcionamento e, posteriormente, quando forem feitas as simulações, "pular" sua execução para resolver a questão mais rapidamente.

Vejamos o método "trocaB":

```
1      private static void trocaB(String a, String b){
2             String tmp = a;
3             a = b;
4             b = tmp;
       }
```

Supondo que a="2" e que b="5", temos o quadro 2, referente ao primeiro teste de mesa.

Quadro 2. Primeiro teste de mesa.

Variáveis / Linha	String a	String b	String tmp
1	"2"	"5"	-
2	"2"	"5"	"2"
3	"5"	"5"	"2"
4	"5"	"2"	"2"

O método "trocaB" troca os valores das variáveis a e b, muda o valor de a em b e muda o valor de b em a. Contudo, essa troca não tem utilidade: na linguagem Java, objetos do tipo String são imutáveis e, ao final da função, as variáveis a e b continuam com os mesmos valores antigos.

Fazendo o mesmo processo para o método "trocaC", ficamos com:

```
1      private static void trocaC(int array[], String a){
2             for(int x = 0; x < array.length; x++){
3                    array[x] = array[x] * Integer.valueOf(a);
             }
      }
```

Supondo como entrada array={1,2,3,4,5} e a = "2", temos o quadro 3, referente ao segundo teste de mesa.

Quadro 3. Segundo teste de mesa.

Variáveis / Linha	Componentes do array	array.length	String a	int x
1	{1,2,3,4,5}	5	"2"	-
2	{1,2,3,4,5}	5	"2"	0
3	{2,2,3,4,5}	5	"2"	0
2	{2,2,3,4,5}	5	"2"	1
3	{2,4,3,4,5}	5	"2"	1
2	{2,4,3,4,5}	5	"2"	2
3	{2,4,6,4,5}	5	"2"	2
2	{2,4,6,4,5}	5	"2"	3
3	{2,4,6,8,5}	5	"2"	3
2	{2,4,6,8,5}	5	"2"	4
3	{2,4,6,8,10}	5	"2"	4
2	{2,4,6,8,10}	5	"2"	5

Assim, o método "trocaC" multiplica os valores contidos no array pelo valor da variável a (no caso, 2).

Podemos, então, avaliar o método "main", conforme segue.

```
      public static void main(String[] args){
1           int array[] = {1,2,3,4,5};
2           String a = "2", b = "5";
3           trocaB(a, b);
4           trocaC(array, a);
5           System.out.print(a + " " + b + " ");
6           for(int x = 0; x < array.length; x++){
7                 System.out.print(array[x] + " ");
8           }
      }
```

Não é mais necessário prosseguir com os testes de mesa: o funcionamento do programa ficou claro. Nas linhas 1 e 2, temos a declaração das variáveis e, na linha, 3 temos uma função que não realiza efeito nenhum. Na linha 4, multiplica-se o valor do array por 2. Na linha 5, imprimem-se os valores de a e b (2 5). Nas linhas 6 e 7, imprimem-se os valores do array (2 4 6 8 10).

Logo, a saída do programa é 2 5 2 4 6 8 10.

Alternativa correta: B.

3. Indicação bibliográfica

- SCHILDT, H.; SKRIEN, D. *Programação com Java – uma introdução abrangente*. São Paulo: McGraw-Hill, 2013.

Questão 2

Questão 2.[2]

Suponha que, para armazenar exatamente 999999 chaves de um índice, um profissional da área da computação tenha escolhido a estrutura de uma árvore B, de grau mínimo 5, com todos os nós completos. Nessa situação, a profundidade dessa árvore é igual a

A. 4

B. 5.

C. 6.

D. 7.

E. 8.

1. Introdução teórica

Árvores B

Um profissional da área da computação, ao projetar um novo sistema, pode deparar com dificuldades na construção de um programa que consuma a memória eficientemente. Isso porque programas que manipulam grande quantidade de dados podem requerer grande quantidade de memória.

Apesar de os preços das memórias terem baixado, as aplicações lidam com uma quantidade cada vez maior de dados, vindos de mais e mais sistemas de informação. Assim, se a disponibilidade de memória aumenta, a quantidade de dados que manipulamos também aumenta de forma notável. Isso significa que não importa o quão grande seja a memória disponível, a quantidade de dados a serem manipulados provavelmente será muito maior.

Lidar com quantidades enormes de dados requer muito cuidado. É possível criar algoritmos que trabalhem de forma ineficiente com grande quantidade de dados, mas que levem à lentidão na operação do sistema.

O engenheiro de computação deve saber utilizar todas as tecnologias disponíveis da maneira mais eficiente possível e deve levar em conta que, nos dias atuais, temos uma hierarquia de armazenamento de dados, conforme descrição a seguir.

O microprocessador é capaz de acessar os seguintes dispositivos: registradores internos (o acesso mais rápido), memórias *cache* (em seus vários níveis), memória RAM principal e, finalmente, armazenamento secundário, geralmente lento e grande. Essa

[2]Questão 32 – Enade 2014 (com adaptações).

característica é bastante nítida no disco rígido, pois, ainda que o armazenamento em dispositivos de estado sólido esteja ganhando espaço, há custo elevado por *byte* armazenado.

Assim, o computador é organizado de acordo com a informação armazenada, em longo prazo, nos meios de armazenamento secundários. A informação é copiada para a memória principal à medida que vai sendo requisitada pelos programas em execução. Quando uma informação não for mais necessária, ela pode ser removida da memória principal. Caso essa informação seja alterada na memória principal, ela deve ser posteriormente copiada para a memória secundária.

Essa abordagem funciona bem em programas que lidam com quantidades pequenas ou moderadas de dados, ou seja, massas de dados que caibam na memória principal.

Contudo, o que fazer caso um programa manipule uma quantidade muito grande de dados, maior do que a quantidade de memória principal disponível? Uma das formas de resolução desse problema é utilizar a memória secundária diretamente pelo processo em execução. Esse processo deve ser otimizado ao máximo, especialmente quando a memória secundária for um disco rígido, com tempos de acesso extremamente elevados.

Uma das formas de otimizar o acesso à memória secundária é empregar uma estrutura de dados que seja especialmente projetada para minimizar o acesso ao disco. Uma dessas estruturas é a chamada árvore B. Antes de definirmos o que é uma árvore B, vamos observar algumas definições básicas válidas para todos os tipos de árvores computacionais.

Uma árvore computacional apresenta um nó inicial, o nó raiz, que é a base da árvore. Cada nó (inclusive o nó raiz) pode ter um conjunto de nós filhos, ainda que diferentes tipos de árvores possam ter restrições em relação ao número de filhos. Cada "geração" de filhos estabelece um nível da árvore.

Os nós no fim da árvore são chamados de nós terminais ou nós folhas.

A profundidade de uma árvore (ou altura de uma árvore) é definida como o número de nós no caminho mais longo da raiz até um nó folha.

A figura 1 apresenta um exemplo de árvore computacional.

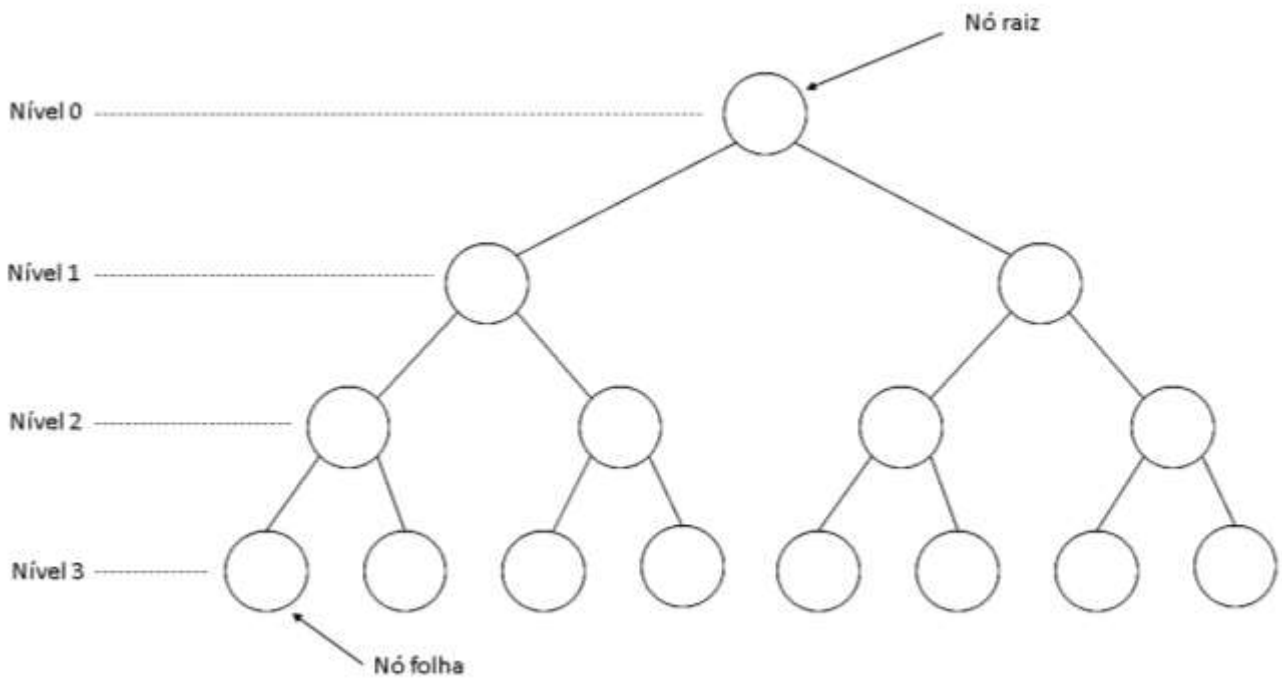


Figura 1. Exemplo de árvore computacional.

A árvore B é projetada para ter altura pequena, com grande quantidade de ramos por nós, diferenciando-se de uma árvore binária, com no máximo dois ramos para cada nó. É importante destacar que cada nó tem uma quantidade mínima e uma quantidade máxima de nós filhos e certa quantidade de chaves para os nós filhos. Essas chaves são organizadas com o intuito de acelerar o acesso às informações armazenadas nos nós filhos. Cada nó de uma árvore B deve ter k registros e k+1 chaves para outros nós.

Essa estrutura é eficiente porque está ligada à forma como os discos rígidos acessam a informação: eles são compostos por uma pilha de discos magnetizados, que rodam com velocidade constante. Além disso, eles dispõem de um pequeno braço mecânico, que também é movido. O disco é dividido em setores e, para recuperar ou alterar dados gravados nesses setores, a cabeça de leitura, contida no braço mecânico, deve ser posicionada cuidadosamente sobre a região onde o dado está armazenado. Esse processo pode parecer rápido para um ser humano, com duração de alguns milissegundos, mas, do ponto de vista eletrônico, esse processo é muito mais lento do que o acesso de dispositivos como memórias. Uma vez que a cabeça de leitura tenha sido corretamente posicionada no disco, a leitura de dados contíguos é muito mais rápida.

Para o sistema ser eficiente, ele deve aproveitar ao máximo a leitura dos dados quando a cabeça já estiver posicionada. Isso é feito por meio da leitura de páginas de dados, de modo que grande quantidade de dados deve ser recuperada a cada leitura do disco. A árvore B foi projetada para possibilitar essa forma de recuperação de dados.

2. Resolução da questão

Para uma árvore B de grau mínimo 5, temos no máximo 10 chaves por nó. O número de registros por nó é de 10-1=9 registros. Com base nessas informações, construímos a figura 1 a seguir.

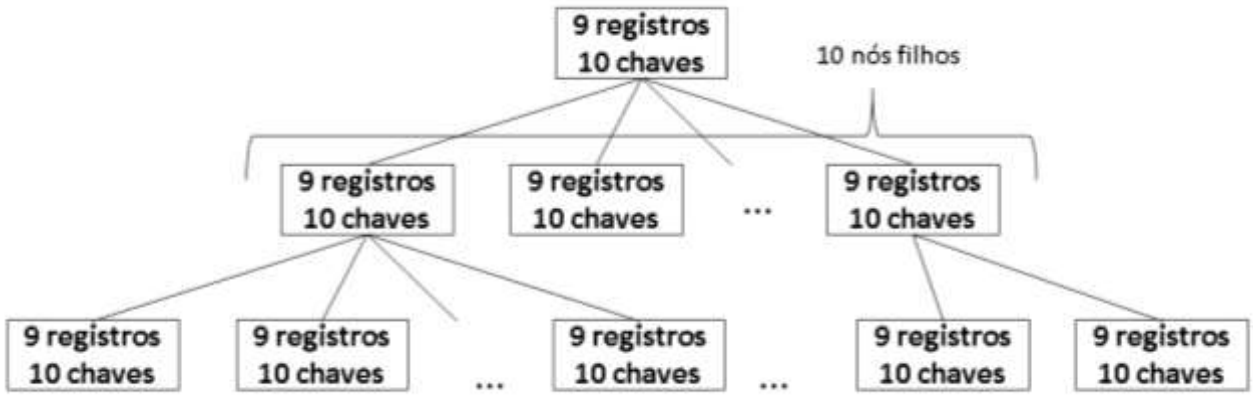


Figura 2. Exemplo de árvore B.

No nível zero dessa árvore (a raiz), temos um nó com 9 registros e 10 chaves. No primeiro nível, temos, para cada uma das chaves, 9 registros e 10 chaves, com um total de 100 chaves e 90 registros. No segundo nível, temos 100 nós e 900 registros. No terceiro nível, temos 1000 nós e 9000 registros. No quarto nível, temos 10000 nós e 90000 registros. No quinto nível, temos 100000 nós e 900000 registros. Somando o total de registros, obtemos o valor 999999, que é o total indicado no enunciado.

Logo, essa árvore apresenta profundidade 5.

Alternativa correta: B.

3. Indicações bibliográficas

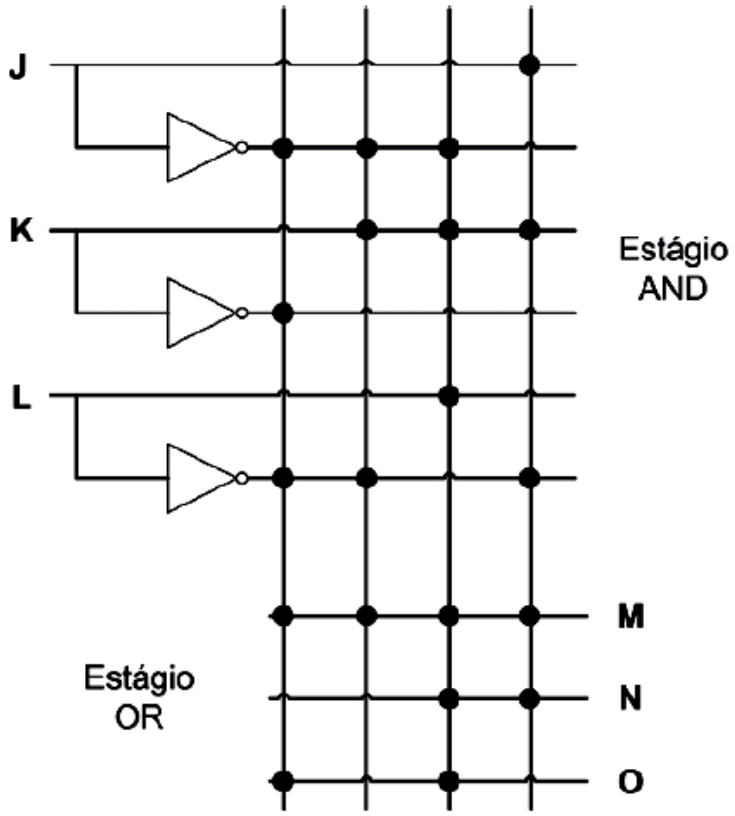
- BROOKSHEAR, J. G. *Ciência da computação: uma visão abrangente.* 11. ed. Porto Alegre: Bookman, 2013.
- CORMEN, T.; LEISERSON, C.; RIVEST, R.; STEIN *Algoritmos - teoria e prática.* São Paulo: Campus, 2012.

Questão 3

Questão 3.[3]

Um componente bastante usado em circuitos lógicos é a matriz lógica programável (ou PLA, do inglês *Programmable Logic Array*). Uma PLA usa como entrada um conjunto de sinais e os complementos desses sinais (que podem ser implementados por um conjunto de inversores). A lógica é implementada a partir de dois estágios: o primeiro é uma matriz de portas AND, que formam o conjunto de termos-produto (também chamados de *mintermos*); o segundo estágio é uma matriz de portas OR, cada uma efetuando uma soma lógica de qualquer quantidade dos *mintermos*. Cada um dos *mintermos* pode ser o resultado do produto lógico de qualquer dos dois sinais de entrada ou de seus complementos.

É comum, em lugar de desenhar todas as portas lógicas de cada um dos estágios, representar apenas a posição das portas lógicas em uma matriz, conforme ilustra a figura a seguir.



[3]Questão 23 – Enade 2014.

A partir da figura apresentada, infere-se que as entradas JKL=000 e JKL=101 levam a saídas MNO iguais, respectivamente, a

A. 000 e 000.
B. 000 e 010.
C. 100 e 101.
D. 101 e 000.
E. 101 e 010.

1. Introdução teórica

1.1. Lógica digital

Dizemos que uma mensagem está codificada de forma digital se ela é codificada em um número finito de símbolos (LATHI, 1998). Especialmente quando utilizamos apenas dois símbolos (0 ou 1, verdadeiro ou falso, aberto ou fechado), dizemos que estamos usando uma codificação binária.

O matemático George Boole (1815-1864) foi o responsável pelo desenvolvimento da álgebra booleana, em seu livro *An Investigation of the Laws of Thought*, de 1854. Contudo, ainda que o desenvolvimento moderno e formal da álgebra booleana seja atribuído a Boole (daí o nome *booleano*), a ideia da utilização de símbolos binários é, provavelmente, muito mais antiga. O matemático Gottfried Leibniz já havia proposto um sistema binário ainda no século XVII, baseado em pesquisas sobre filosofia oriental. Existem evidências de emprego de sistemas binários em antigas civilizações da Ásia e da África.

Ao utilizarmos a álgebra booleana, podemos descrever as saídas de um circuito lógico em função das suas entradas por meio de equações ou expressões booleanas (TOCCI e WIDMER, 2000).

Sistemas digitais são sistemas eletrônicos que podem ser projetados com técnicas baseadas na álgebra booleana. Além disso, esse tipo de sistema apresenta uma série de vantagens intrínsecas em relação aos sistemas eletrônicos analógicos, especialmente em relação à robustez e ao ruído. Informações transmitidas de forma digital têm elevado grau de imunidade ao ruído e, mesmo no caso de algum erro de transmissão, existe a possibilidade da utilização de protocolos que podem corrigir esses erros.

Uma vez que tivermos uma expressão booleana para um sistema, podemos escrever todas as possíveis combinações de entradas e saídas e, dessa forma, podemos antecipar todos os possíveis estados do sistema.

1.2. Tabelas verdade

Para expressar o resultado de funções lógicas, utilizamos tabelas verdade, representações tabulares de uma expressão lógica que mostram todas as possíveis entradas da expressão e os resultados (ou saídas) dessas entradas. Por meio dessas tabelas, podemos inspecionar e entender o funcionamento de uma porta lógica, por exemplo.

O número de linhas de uma tabela verdade depende, essencialmente, do número das variáveis de entrada, uma vez que cada variável pode apenas ter o valor 0 ou o valor 1. Assim, para uma variável, temos apenas duas possibilidades; para duas variáveis, temos quatro possibilidades; para três variáveis, temos oito possibilidades. Dessa forma, para n variáveis, temos 2^n possibilidades ou linhas na tabela verdade.

Como o número de linhas da tabela cresce de forma exponencial com o número de variáveis de entrada, a utilização de tabelas da verdade em circuitos digitais reais é limitada apenas a pequenos circuitos. Por exemplo, uma expressão digital com apenas 10 variáveis (n=10) iria requerer uma tabela com 2^{10} = 1024 linhas, o que evidencia a limitação dessa técnica.

1.3. Portas lógicas básicas

Para a construção de circuitos digitais, utilizam-se dispositivos chamados portas lógicas. Esses dispositivos, normalmente encontrados na forma de circuitos integrados, apresentam funções booleanas básicas já prontas, como, por exemplo, as funções OU, E ou OU EXCLUSIVO.

Dessa forma, o projetista de circuitos digitais não precisa se preocupar em desenvolver essas estruturas básicas: apenas deve pensar em como interligar essas portas com o objetivo de obter determinada função booleana potencialmente complexa.

1.3.1. Porta lógica "E"

A porta lógica E (chamada no inglês de AND) é representada pelo símbolo ilustrado na figura 1.

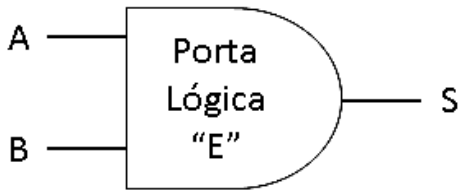


Figura 1. A porta lógica "E".

Sua tabela verdade é dada na tabela 1.

Tabela 1. Tabela verdade da porta lógica "E".

A (entrada)	B (entrada)	S(saída)
0	0	0
0	1	0
1	0	0
1	1	1

Se quisermos construir um circuito lógico com mais entradas, podemos conectar várias portas lógicas entre si, como mostrado na figura 2.

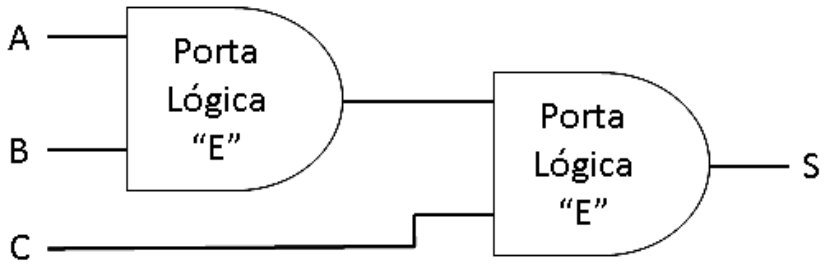


Figura 2. Conectando várias portas lógicas "E".

Observe que, nesse caso, a tabela da verdade também deve conter a variável de entrada C, e passa a ter 8 linhas, conforme tabela 2 a seguir.

Tabela 2. Tabela-verdade da figura 2.

A (entrada)	B (entrada)	C (entrada)	S (saída)
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

Podemos agrupar as variáveis de entrada, de forma a obtermos uma notação mais compacta do diagrama, de acordo com a figura 3.

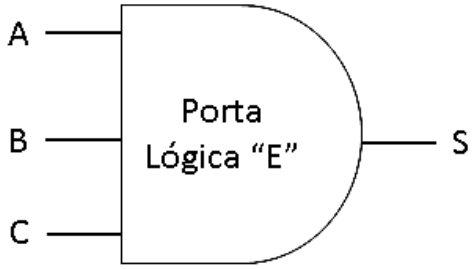


Figura 3. Uma porta lógica "E" com várias portas.

O resultado final é o mesmo, mas a notação é mais compacta e permite que se façam diagramas maiores e não muito poluídos. Além disso, ao adicionarmos um pequeno círculo na saída de uma porta lógica, estamos invertendo a saída dessa porta. Por exemplo, se fizermos isso com a porta lógica anterior, obtemos a situação da figura 4.

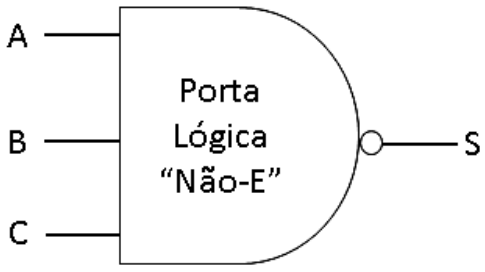


Figura 4. A porta lógica "NÃO-E".

Temos, como resultado, a inversão do bit de saída da porta E e obtemos a porta lógica NÃO-E (NAND), que tem como tabela verdade a tabela 3 que segue.

Tabela 3. Tabela-verdade da porta lógica da figura 4.

A (entrada)	B (entrada)	C (entrada)	S (saída)
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

Observe a tabela 3 é apenas a tabela da porta lógica "E" com os bits de saída invertidos. Esse procedimento (de adicionar um pequeno círculo na saída da porta lógica para fazer a inversão) é válido para todas as portas lógicas, e não apenas para a porta lógica "E".

1.3.2. Porta lógica "OU"

A porta lógica OU (chamada no inglês de OR) está representada na figura 5.

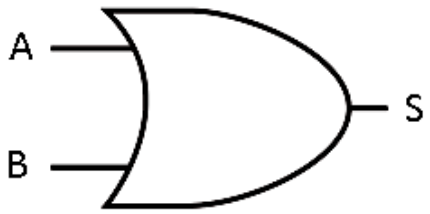


Figura 5. A porta lógica "OU".

A tabela-verdade da porta lógica "OU" é dada pela tabela 4 a seguir.

Tabela 4. Tabela da verdade da figura 5.

A (entrada)	B (entrada)	S(saída)
0	0	0
0	1	1
1	0	1
1	1	1

1.3.3. Porta lógica "OU EXCLUSIVO"

A porta lógica "OU EXCLUSIVO" (chamada no inglês de XOR) está representada na figura 6.

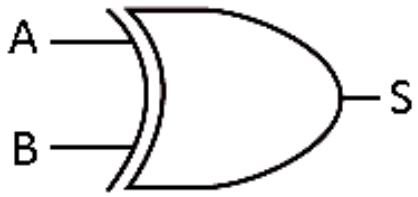


Figura 6. A porta lógica "NÃO-OU".

Sua tabela-verdade é dada pela tabela 5 a seguir.

Tabela 5. Tabela da verdade da figura 6.

A (entrada)	B (entrada)	S(saída)
0	0	0
0	1	1
1	0	1
1	1	0

2. Resolução da questão

Para resolvermos a questão, precisamos identificar as expressões booleanas das variáveis M, N e O.

Inicialmente, identificamos os termos do estágio AND, como indicado na figura 7 a seguir.

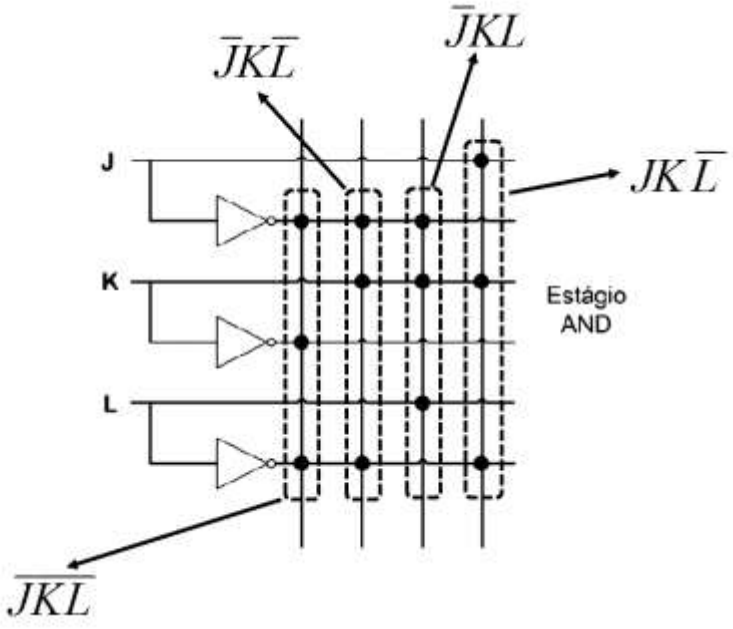


Figura 7. Termos do estágio AND.

Finalmente, devemos compor os termos apresentados para cada uma das linhas das variáveis M, N e O, lembrando que esse é o estágio do OU:

$$M = \overline{JKL} + \bar{J}K\bar{L} + \bar{J}KL + JK\bar{L}$$

$$N = \bar{J}KL + JK\bar{L}$$

$$O = \overline{JKL} + \bar{J}KL$$

Substituindo J=0, K=0 e L=0 nas expressões, ficamos com:

- M=111+101+100+001=1+0+0+0=1
- N=100+001=0+0=0
- O=111+100=1+0=1

Logo MNO=101.

Substituindo J=1, K=0 e L=1 nas expressões, ficamos com:

- M=010+000+001+100=0+0+0+0=0
- N=001+100=0
- O=000+001=0

Logo MNO=000.

Alternativa correta: D.

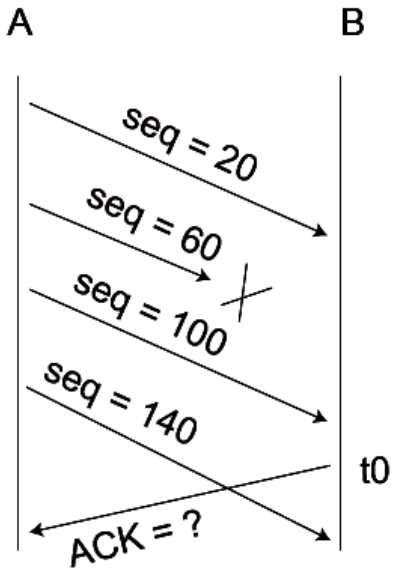
3. Indicações bibliográficas

- LATHI, B. P. *Modern digital and analog communication systems*. 3. ed. New York: Oxford University Press, 1998.
- RAM, B. *Computer fundamentals: architecture and organization*. Nova Delhi: New Age International, 2000.
- ROSEN, K. *Discrete Mathematics and Its Applications*. 7. ed. Porto Alegre: McGraw-Hill, 2011.
- TOCCI, R. J.; WIDMER, N.S. *Sistemas digitais: princípios e aplicações*. 7. ed. Rio de Janeiro: LTC, 2000.

Questão 4

Questão 4.[4]

Acerca do protocolo de transporte TCP (*Transmission Control Protocol*) utilizado na Internet, considere o esquema abaixo, que mostra a comunicação entre dois processos A e B. No diagrama, o tempo cresce de cima para baixo e as setas diagonais representam segmentos TCP enviados de A para B ou de B para A, dependendo da orientação da seta. Os números de sequência dos dados de aplicação enviados de A para B estão indicados sobre as setas. O processo A enviou segmentos com 40 bytes de dados de aplicação para B. O número de sequência do primeiro byte enviado através da conexão de A para B foi 20. Dos quatro segmentos enviados de A para B, o segundo segmento foi perdido pela rede e não alcançou o destino.



Com base na situação descrita acima, o número de confirmação (ACK) enviado pelo TCP de B para A, no instante de tempo t0, é igual a

A 20.

B. 59.

C. 60.

D. 100.

E. 140.

[4]Questão 24 – Enade 2014.

1. Introdução teórica

Protocolo de Controle de Transmissão (TCP)

Para que ocorra a comunicação adequada entre computadores, é necessária a organização das informações enviadas pela rede. Isso é feito por meio da utilização de um protocolo de comunicação, que estabelece uma forma padrão para transmissão e recebimento de dados entre máquinas.

O Protocolo de Controle de Transmissão (TCP) corresponde à camada 4 do modelo Open System Interconnection (OSI), conhecida como camada de transporte. Ele está associado a outro protocolo importante, o Internet Protocol (IP).

O protocolo TCP é orientado à conexão feita pelos sockets.

Os sockets são entidades virtuais e existem tanto no servidor quanto no cliente. Eles são compostos por dois números: o endereço IP e uma porta. A conexão é sempre estabelecida entre um par de computadores e cada um dos computadores deve ter um endereço IP e uma porta associada. Por isso, dizemos que o protocolo TCP é um protocolo ponto-a-ponto (TANENBAUM e WETHERALL, 2010).

Outra característica importante do protocolo IP é que ele é full duplex. Isso significa que é possível enviar e receber dados ao mesmo tempo (TANENBAUM e WETHERALL, 2010).

A comunicação é dividida em segmentos, cada um com um cabeçalho associado a certa quantidade de dados. Nesse cabeçalho, há uma série de campos especiais, como o número de sequência e o número de confirmação, além de campos como checksums e diversas flags de controle.

Um dos propósitos desses campos é garantir a entrega ordenada de pacotes, além de garantir a confiabilidade. Em uma rede de computadores, seja cabeada ou sem fio, pode ocorrer a perda de pacotes de informações ou a mudança na ordem de chegada dos pacotes no receptor. Com isso, devido a diversos fatores (como, por exemplo, diferentes rotas para o envio de dados), um conjunto de pacotes pode ser enviado em uma sequência e pode ser recebido em uma sequência diferente.

Por meio dos números de sequência, o protocolo TCP pode reconstruir a ordem original de envio no receptor, mesmo que a ordem de recebimento tenha sido alterada pela transmissão.

Além disso, campos como o de checksum, por exemplo, permitem a identificação de erros nos dados recebidos. As máquinas envolvidas podem, então, solicitar a retransmissão de pacotes perdidos ou de pacotes recebidos com erros, com aumento do grau de confiabilidade da comunicação.

2. Resolução da questão

Sabemos que o processo B está enviando um pacote ACK relativo ao recebimento do primeiro pacote, com número de sequência igual a 20.

Além disso, sabemos que a quantidade de dados enviados no pacote foi de 40 bytes.

Dessa forma, o número ACK enviado de B para A deve ser igual ao número de sequência do pacote somado à quantidade de dados em bytes, ou seja, 40+20=60 bytes.

Alternativa correta: C

3. Indicação bibliográfica

- TANENBAUM, A. S.; WHETHERALL, D. J. *Redes de computadores.* São Paulo: Pearson Prentice Hall, 2011.

Questão 5

Questão 5.[5]

A criação dos serviços em nuvens (clouds) teve como consequência o fato de as tarefas de processamento (como ferramentas para edição de documentos), armazenamento de dados (como arquivos e documentos) e mensagens (webmail) deixarem de ser executadas em estações cliente locais sem conexão à rede e passarem a ser delegadas a equipamentos remotos conectados através da Internet.

Cada vez mais surgem empresas que oferecem nuvens de equipamentos conectados através da Internet, com clusters de equipamentos e redundância em múltiplos sites para a prestação terceirizada desse tipo de serviço, de modo a oferecer maior desempenho e disponibilidade. Por outro lado, aumentam os riscos de quebra da privacidade dos dados armazenados.

Nesse contexto de mudança de um sistema local para a adoção de serviços em nuvens, responda as questões a seguir.

A. Como mudam os requisitos da plataforma do cliente?
B. Que requisitos devem ser atendidos pela infraestrutura local de rede e telecomunicações?
C. Como esse tipo de serviço pode apresentar melhor disponibilidade e menor risco de perda de dados?
D. Que benefícios são esperados com a adoção de serviços em nuvens?

1. Introdução teórica

1.1. Sistemas distribuídos

Desde o advento dos computadores, das redes de computadores e dos sistemas operacionais capazes de executar mais de um processo simultaneamente (sistemas operacionais multitarefa) e de atender a mais de um usuário simultaneamente (sistemas multiusuários), surgiu a possibilidade do compartilhamento de recursos não apenas localmente, mas também remotamente, por meio da rede.

A literatura apresenta diversas definições de sistema distribuído.

Veríssimo e Rodrigues (2001) definem o sistema distribuído como

[5]Questão Discursiva 4 – Enade 2014.

> *um sistema composto de vários computadores que se comunicam através de uma rede, hospedando processos que usam um conjunto comum de protocolos distribuídos para auxiliar na execução coerente de atividades distribuídas.*

Para Coulouris et al (2012), um sistema distribuído é aquele

> *em que os componentes localizados em uma rede de computadores comunicam-se e coordenam suas ações somente por meio da troca de mensagens.*

Essas definições retratam duas características fundamentais dos sistemas distribuídos:

- os processos que são executados em várias máquinas diferentes simultaneamente e se comunicam por troca de mensagens (transportadas pela rede);
- os recursos utilizados podem encontrar-se em lugares diferentes daqueles onde estão os processos que os utilizam.

1.2. Computação em nuvem

Hassan (2011) define a computação em nuvem como

> *um modelo de computação sob demanda, composto de recursos de tecnologia da informação (tanto hardware quanto software) autônomos e interconectados. Os provedores desses serviços devem oferecer nuvens com níveis pré-definidos de qualidade de serviço (QoS), pela Internet como um conjunto de serviços fáceis de serem utilizados, escaláveis e baratos para clientes interessados, comercializado por meio de assinaturas.*

Dessa forma, encaramos a computação em nuvem como um tipo moderno de sistema distribuído, que utiliza clientes com boa capacidade de processamento (como notebooks ou telefones celulares) e a internet como meio de conexão.

As vantagens da computação em nuvem são

- redução de custos operacionais;
- diminuição de custos de manutenção;
- aumento da escalabilidade.

Há riscos envolvidos na utilização de serviços em nuvem, que envolvem

- segurança, especialmente com relação aos dados da empresa que estão sendo armazenados em computadores em locais externos à sua sede e em máquinas de terceiros;
- confiabilidade, uma vez que um erro cometido por um terceiro pode tornar os sistemas corporativos indisponíveis.

Devem, também, ser considerados problemas no uso da nuvem em relação à velocidade, à estabilidade e à confiabilidade na conexão à internet e em relação à falta de padrão entre os diversos fornecedores de tecnologias (HASSAN, 2011).

2. Resolução da questão

A. Uma vez que os serviços serão oferecidos pela internet, serão necessárias boa infraestrutura de rede e excelente conexão com a internet, pois será preciso algum mecanismo de redundância, caso ocorra uma falha em um dos fornecedores. Além dos problemas de velocidade e de disponibilidade da comunicação, é necessário que conexão com a internet seja feita com algum mecanismo de criptografia, com elevados padrões de segurança. Visto que os serviços são oferecidos remotamente, a plataforma do cliente pode se tornar mais simples se for utilizado um hardware mais barato e com menor pressão para atualização (HASSAN, 2011).

B. A rede é um dos fatores fundamentais para o sucesso de um projeto que utiliza computação em nuvem. Qualquer problema de comunicação, como lentidão ou indisponibilidade, pode significar a interrupção do fornecimento do serviço. Por isso, a infraestrutura de rede deve ter alto desempenho e alta disponibilidade, ou seja, deve apresentar qualidade de serviço. A presença de mais de um tipo de acesso à internet também é fundamental, para evitar interrupções causadas por problemas no provedor, caso haja apenas um.

C. Como qualquer sistema da área de tecnologia da informação, deve haver um plano de back-up adequado para sua recuperação, caso ocorra algum tipo de falha. Para que haja aumento da disponibilidade, deve haver redundância na conexão com a internet, pois, caso um fornecedor esteja fora do ar, outra conexão deve estar disponível. Entretanto, enquanto um sistema de armazenamento nas nuvens aumenta a segurança contra perdas acidentais, isso também diminui a garantia de confidencialidade dos dados, como, por exemplo, a exposição a ataques de hackers.

D. A adoção de serviços baseados em nuvem pode apresentar vantagens, como a diminuição dos custos dos equipamentos clientes e a redução do custo de manutenção. O sistema também pode ter diferentes tipos de acessos pela utilização de dispositivos como

telefones celulares e *tablets.* Dependendo do tipo da qualidade do provedor escolhido, o sistema pode apresentar maior disponibilidade do que um sistema "tradicional", especialmente se considerarmos o caso de pequenas e médias empresas, nas quais, frequentemente, os sistemas utilizam hardware mais limitado. Um sistema em nuvem de um bom fornecedor pode ter uma infraestrutura muito mais adequada, desde que se tenha suficiente certeza da preservação da confidencialidade dos dados, que, afinal, são parte fundamental do patrimônio imaterial de toda empresa.

3. Indicações bibliográficas

- COULOURIS, G. F.; DOLLIMORE, J.; KINDBERG, T.; BLAIR, G. *Distributed systems: concepts and design.* Harlow: Addison-Wesley, 2012.
- HASSAN, Q. Demystifying cloud computing. *The journal of defense software Engineering*, v. 1, p. 16-21, 2011.
- VERÍSSIMO, P.; RODRIGUES, L. *Distributed systems for system architects.* Boston: Kluwer academic, 2001.

Questão 6

Questão 6.[6]

Leia o texto a seguir.

De todas as propostas do Marco Civil da Internet, uma das mais polêmicas e importantes é a denominada "neutralidade da rede". O Marco Civil defende que não deve haver "pedágios" na internet, ou seja, nenhuma empresa poderá criar barreiras para algum tipo de conteúdo com qualquer tipo de interesse financeiro.
As empresas dizem que a neutralidade total mata a possibilidade de oferecer pacotes mais acessíveis. Os defensores do projeto, por outro lado, dizem que a não aprovação seria uma medida antipopular, que criaria mais exclusão social, impedindo que os mais pobres usem os serviços mais caros.

Disponível em <http://olhardigital.uol.com.br>. Acesso em 20 jul. 2014 (com adaptações).

A partir das informações apresentadas e em relação à "neutralidade da rede", avalie as afirmativas.

I. Com a lei de neutralidade da rede brasileira, o roteamento interno na rede de uma corporação deve tratar todos os protocolos ou serviços de modo igualitário.

II. A mudança de cenário com a adoção da lei da neutralidade da rede é exemplo de como as empresas e profissionais de tecnologia devem estar continuamente se atualizando e estar prontos para readequar seus produtos e serviços aos novos requisitos técnicos e sociais.

III. A lei brasileira da neutralidade da rede permite que um provedor de aceso à Internet, notando que seus usuários usam mais serviços de mensagens instantâneas que de transferências de arquivos, possa aumentar a prioridade do primeiro tráfego em relação ao do segundo para melhorar a satisfação de seus clientes.

É correto o que se afirma em

A. II, apenas.
B. III, apenas.
C. I e II, apenas.
D. I e III, apenas.
E. I, II e III.

[6]Questão 19 – Enade 2014.

1. Introdução teórica

Marco Civil da Internet

O artigo primeiro do Capítulo I da Lei Nº 12.965/2014 estabelece os princípios do Marco Civil da Internet:

> *esta lei estabelece princípios, garantias, direitos e deveres para o uso da internet no Brasil e determina as diretrizes para atuação da União, dos Estados, do Distrito Federal e dos Municípios em relação à matéria.*

O Capítulo III, seção I da Lei Nº 12965/2014 trata da neutralidade da Internet no Marco Civil da Internet, conforme segue.

> *Art. 9º O responsável pela transmissão, comutação ou roteamento tem o dever de tratar de forma isonômica quaisquer pacotes de dados, sem distinção por conteúdo, origem e destino, serviço, terminal ou aplicação.*
> *§1º A discriminação ou degradação do tráfego será regulamentada nos termos das atribuições privativas do Presidente da República previstas no inciso IV do art. 84 da Constituição Federal, para a fiel execução desta Lei, ouvidos o Comitê Gestor da Internet e a Agência Nacional de Telecomunicações, e somente poderá decorrer de:*
> *I - requisitos técnicos indispensáveis à prestação adequada dos serviços e aplicações; e*
> *II - priorização de serviços de emergência.*
> *§2º Na hipótese de discriminação ou degradação do tráfego prevista no § 1o, o responsável mencionado no caput deve:*
> *I - abster-se de causar dano aos usuários, na forma do art. 927 da Lei no 10.406, de 10 de janeiro de 2002 - Código Civil;*
> *II - agir com proporcionalidade, transparência e isonomia;*
> *III - informar previamente de modo transparente, claro e suficientemente descritivo aos seus usuários sobre as práticas de gerenciamento e mitigação de tráfego adotadas, inclusive as relacionadas à segurança da rede; e*
> *IV - oferecer serviços em condições comerciais não discriminatórias e abster-se de praticar condutas anticoncorrenciais.*
> *§3º Na provisão de conexão à internet, onerosa ou gratuita, bem como na transmissão, comutação ou roteamento, é vedado bloquear, monitorar, filtrar ou analisar o conteúdo dos pacotes de dados, respeitado o disposto neste artigo.*
> *BRASIL. Lei Nº 12.965, de 23 de abril de 2014.*
> Disponível em <http://www.planalto.gov.br/ccivil_03/_ato2011-2014/2014/lei/l12965.htm>. Acesso em 17 mai. 2017.

2. Análise das afirmativas

I – Afirmativa incorreta.
JUSTIFICATIVA. O Marco Civil da Internet não regulamenta as redes internas das empresas, apenas a Internet.

II – Afirmativa correta.
JUSTIFICATIVA. As empresas e os profissionais da área de tecnologia devem estar cientes de alterações na legislação que afetem a sua área de atuação e devem implantar as medidas que venham a ser determinadas por essas leis.

III – Afirmativa incorreta.
JUSTIFICATIVA. A priorização de trafego na Internet, deve ocorrer, segundo o Marco Civil da Internet (BRASIL, 2014), apenas nos casos de

> *I - requisitos técnicos indispensáveis à prestação adequada dos serviços e aplicações;*
> *II - priorização de serviços de emergência.*

Logo, a priorização para liberar mais banda para um serviço que exija maior tráfego não está prevista.

Alternativa correta: A.

3. Indicações bibliográficas

- BRASIL. *Lei Nº 12.965, de 23 de abril de 2014*. Disponível em <http://www.planalto.gov.br/ccivil_03/_ato2011-2014/2014/lei/l12965.htm>. Acesso em 17 mai. 2017.
- de JESUS, D.; MILAGRE, J. A. *Marco Civil da Internet – comentários à Lei Nº 12.965/14*. São Paulo: Saraiva, 2014.

Questão 7

Questão 7.[7]

A recursividade, técnica muito utilizada em programação, permite que um número finito de instruções processe um número potencialmente infinito de dados por intermédio do paradigma dividir-para-conquistar.

A função *Interseção (v1v2, v3v4: segmento)* é uma função que retorna a interseção P entre dois segmentos concorrentes v1v2 e v3v4 através das divisões sucessivas do segmento v1v2, como mostra o seguinte pseudocódigo:

```
01 função Intersecao (v1v2, v3v4: segmento): ponto
02   inicio
03    se v1 != v2 então
04      inicio
05        P ← ponto_medio (v1v2);
06        se (pertence(P, v3v4)) retorne P;
07        se lado(v1, v3v4) != lado(P,v3v4) então
08          início
09            P ← Intersecao(v1P, v3v4);
10            retorne P;
11          fim
12        senão
13          início
14            P ← Intersecao(Pv2, v3v4);
15            retorne P;
16          fim
17      fim
18      senão
19        início
20          se (pertence(v1, v3v4)) retorne v1;
21          senão retorne vazio;
22        fim
23   fim
```

A função *ponto_medio(AB)* retorna o ponto médio do segmento AB. A função *pertence(A,BC)* retorna um valor lógico, Verdadeiro ou Falso, quanto à pertinência do ponto A com respeito ao segmento BC. A função *lado(A,BC)* retorna o valor em que se encontra o ponto A em relação ao segmento BC.

A respeito desse pseudocódigo, avalie as afirmativas a seguir.

I. v3v4 é dividido sucessivamente para conquistar P.

II. As condições de parada da recursão estão nas linhas 6, 20 e 21.

III. As condições de parada da recursão estão nas linhas 6, 10, 15, 20 e 21.

[7]Questão 30 – Enade 2014.

IV. Pelas condições iniciais da posição relativa entre os segmentos v1v2 e v3v4, a linha 21 nunca será executada.

É correto apenas o que se afirma em

A. I e II.

B. II e IV.

C. III e IV.

D. I, II e III.

E. I, III e IV.

1. Introdução teórica

Recursividade

Uma sub-rotina é dita recursiva quando chama a si mesma. Como exemplo, considere a função do trecho de programa a seguir, escrito em português estruturado, que calcula o fatorial de um número (x) dado.

```
1     função fatorial(x: inteiro): inteiro
2     inicio
3        se x = 0 então
4           fatorial = 1;
5        senão
6           fatorial = x*fatorial(x-1);
7        fim_se;
8     fim.
```

Na primeira linha, realizamos a declaração da função, que recebe x como valor de entrada e retorna um valor inteiro.

Nas linhas 3 e 4, se o valor dado para calcular o fatorial for 0, é retornado o valor 1, pois 0!=1. Caso contrário, é executada a linha 6, que faz uma nova chamada da função fatorial, mas com o valor de entrada diminuído de uma unidade. O cálculo é feito com sucessivas chamadas da função fatorial. Por exemplo, o cálculo de 5! por esse algoritmo seria:

$$5! = 5 \times 4! = 5 \times 4 \times 3! = 5 \times 4 \times 3 \times 2! = 5 \times 4 \times 3 \times 2 \times 1 \times 1$$

A cada fatorial representado no cálculo, é feita uma nova chamada da função fatorial. Na última passagem, o 1 final é o resultado da chamada da função com x=0.

Um método importante em programação é o da bissecção, também chamado de "dividir para conquistar" (*divide and conquer*). Nesse método, começamos com um intervalo de valores [a,b], como o do segmento de reta mostrado na figura 1, e calculamos o valor médio desse segmento, que, no caso, é P.

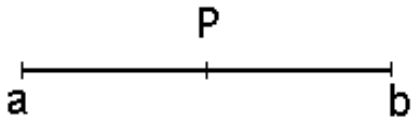


Figura 1. Segmento de reta com os pontos **a** e **b**.

Testamos se P satisfaz à condição do problema (no caso, se intercepta v3v4). Se essa condição não for satisfeita, precisamos decidir qual dos segmentos será usado em seguida, a-P ou P-b. Suponha que decidamos por a-P. Então, esse segmento é passado como entrada do método e, nesse caso, será obtido o ponto médio entre a e P, chegando-se a P', como mostrado na figura 2.

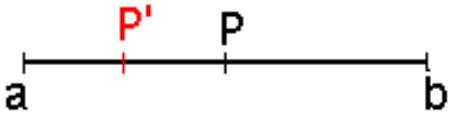


Figura 2. Ponto P' no segmento de reta com os pontos **a** e **b**.

Testamos, novamente, a condição com P'. Se essa condição não for satisfeita, precisamos decidir qual dos segmentos será usado em seguida, a-P' ou P'-P. Suponha que decidamos por P'-P. Então, esse segmento é passado como entrada do método e, nesse caso, ele é dividido ao meio e chegamos a P'' (figura 3).

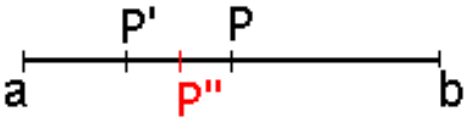


Figura 3. Ponto P'' no segmento de reta com os pontos **a** e **b**.

Prosseguimos assim sucessivamente, até obtermos um ponto $P^{(n)}$ que satisfaça a condição dada pelo problema.

Esse tipo de algoritmo é usado tanto para a resolução do problema apresentado quanto, por exemplo, para:

- a obtenção da intersecção de duas retas;
- o cálculo do ponto de máximo e do ponto de mínimo de uma função, caso existam;
- a determinação das raízes de funções;
- a identificação da ocorrência em uma lista ordenada.

Vale acrescentar que uma reta é um segmento composto por pontos de forma ordenada (coordenadas sempre estritamente crescentes ou decrescentes).

2. Análise das afirmativas

I – Afirmativa incorreta.
JUSTIFICATIVA. Na linha 5, vemos que o ponto P passa a ser o ponto médio do segmento v1v2. Nas linhas 09 e 14, a função intersecção é chamada novamente (há recursão), mas passando, como valores de entrada, (v1P, v3v4) e (Pv2, v3v4), respectivamente. Note que no primeiro caso, v2 é substituído pelo ponto médio do segmento v1v2, e, no segundo caso, v1 é substituído pelo ponto médio de v1v2. Em ambos os casos, nada é feito com o segmento v3v4.

II – Afirmativa correta.
JUSTIFICATIVA. As condições de parada da recursão estão nas linhas 6, 20 e 21, pois, nelas, são feitos os testes e a tomada a decisão de continuar ou não com o *looping*.

III – Afirmativa incorreta.
JUSTIFICATIVA. Na linha 6, temos um condicional que impõe que, se P pertence a v3v4, o valor P é retornado. Nesse caso, após sucessivas divisões do segmento v1v2, obteve-se um ponto P que pertence ao segmento v3v4. Logo, encontrou-se o ponto de intersecção dos dois segmentos, valor contido na variável P.
Na linha 20, temos o que é feito caso o condicional da linha 3 não seja satisfeito. Nesse caso, se a condição $v1 \neq v2$ não é satisfeita, não temos um segmento de reta v1v2, mas apenas um ponto resultante das bissecções. Na linha 20, é testado se esse ponto faz parte do segmento v1v3 e, portanto, é ponto de intersecção. Na linha 21, o programa retorna 'vazio' caso: nessa situação, v1=v2 não pertence ao seguimento v3v4.
Nas linhas 10 e 15, o programa não é interrompido: continua sendo executado o *looping* da linha 7 enquanto a condição dada nessa linha for satisfeita. Logo, essas linhas não apresentam condições de parada de recursão.

IV – Afirmativa correta.

JUSTIFICATIVA. Se os segmentos de reta são concorrentes, então existe um ponto que é intersecção dos segmentos v1v2 e v3v4. A linha 21 é executada caso a condição da linha 20 não seja satisfeita, ou seja, se v1 pertence a v3v4. Note que o bloco das linhas 19 a 22 é executado quando a condição da linha 3 não é satisfeita, ou seja, quando v1=v2 após sucessivas bissecções.

Alternativa correta: B.

3. Indicações bibliográficas

- ASCENCIO, A. F. G.; CAMPOS, E. A. V. *Fundamentos de programação de computadores*. São Paulo: Prentice Hall, 2002.
- FORBELLONE, A. L. V.; EBERSPACHER, H. F. *Lógica de programação*. São Paulo: Makron Books, 2005.

Questão 8

Questão 8.[8]

Leia o texto a seguir.

Os impactos relativos ao comprometimento da qualidade ambiental, durante a fase de implantação de um empreendimento, considerando-se especificamente a poluição sonora, dependem basicamente dos níveis de emissão de ruído e das distâncias dos receptores em relação às fontes de emissão da área em análise. Um equipamento que emita um valor medido de 95 dB (A), a cerca de 1,5 metro de distância, apresentará a diminuição de ruído conforme indicado no quadro a seguir.

Diminuição do ruído com a distância

Distância em metros	1,5	3,0	6,0	12,0	24,0	48,0
Nível sonoro dB (A)	95	89	83	77	71	65

Estudo de Viabilidade Ambiental (EVA). Volume III – Avaliação de Impactos. Secretaria Municipal de Desenvolvimento Econômico e do Trabalho de São Paulo. São Paulo. 2011. Disponível em <http://www.prefeitura.sp.gov.br>. Acesso em 31 jul. 2014 (com adaptações).

Abaixo é apresentado um trecho de um programa de computador, escrito em pseudocódigo, que armazena, na matriz M, valores oriundos do quadro.

```
...
i <- 2
para j de 1 ate 3 passo 1 faça
  M[i, j] <- 95
  M[i, j+1] <- 89
  M[i, j+2] <- 83
  M[i, j+3] <- 77
fimpara
...
```

Com base na simulação da resposta apresentada após a execução desse trecho de programa, assinale a opção que contempla corretamente os respectivos valores armazenados na matriz M.

A.

Matriz	1	2	3	4	5	6
1						
2	95	95	95	89	83	77

B.

Matriz	1	2	3	4	5	6
1						
2	95	89	83	77	71	65

C.

Matriz	1	2	3	4	5	6
1						
2	65	71	77	83	89	95

[8]Questão 31 – Enade 2014.

D.

Matriz	1	2	3	4	5	6
1						
2	95	89	83	77		

E.

Matriz	1	2	3	4	5	6
1						
2			95	89	83	77

1. Introdução teórica

1.1. Lógica de programação

A questão apresenta um trecho de um programa escrito em pseudocódigo. O exercício, ao não se adotar uma linguagem de programação específica, coloca seu foco no entendimento da lógica de programação. As instruções escritas em pseudocódigo estão em português. Um exemplo de linguagem de programação em pseudocódigo é o Português Estruturado.

Nesse trecho de código, temos dois elementos: uma matriz e uma estrutura de repetição. Podemos observar que a matriz é usada para armazenar os dados, e isso é feito com o uso de uma estrutura de repetição.

1.2. Matrizes

Uma variável é uma posição na memória do computador usada para armazenar um dado ao longo da execução de um programa. Em função dos conteúdos que irão armazenar, as variáveis podem ser de diversos tipos. Existem, por exemplo, as variáveis do tipo inteiro, para a armazenagem de números sem decimais, e as variáveis do tipo real, para a armazenagem de números com decimais.

Quando armazenamos um dado em uma variável, dizemos que estamos atribuindo um valor a essa variável. Isso pode ser feito com diversos sinais, como, por exemplo, =, <-, ou ≔.

Dessa forma, a linha de programa contendo A<-3 indica que a variável A recebe e passa a armazenar o valor 3. Não estamos usando ponto e vírgula no final da linha como seria feito em português estruturado, pois isso não é feito no pseudocódigo do enunciado.

Se encadearmos uma sequência de variáveis, temos uma *string*. Para determinarmos a posição na *string*, ou seja, qual variável da sequência vamos acessar, usamos um contador. Nesse contexto, a *string* dada por *nome[i]* indica a posição i da *string*. Então, *nome[2]<-3* representa a segunda posição da *string* recebendo como conteúdo o número 3.

Podemos, também, combinar uma sequência de *strings* para formar uma matriz. Na matemática, cada elemento de uma matriz é designado por uma posição i,j, em que i é a linha e j a coluna desse elemento. Em programação, seguimos a mesma ideia: cada elemento da matriz é designado por sua linha e por sua coluna. Logo, precisamos de dois ponteiros para identificarmos um elemento em uma matriz. Usamos, então, a notação a seguir para um elemento de uma matriz:

nome [i,j]

Nessa notação, *i* representa a linha da matriz onde o elemento se encontra e *j* a coluna.

Então, *nome [1,2]<-4* indica que o elemento da primeira linha e segunda coluna da matriz recebe e passa a armazenar o valor 4.

1.3. Estruturas de repetição

Estruturas de repetição são recursos de programação para se repetir a execução de dado bloco do programa enquanto dada condição for satisfeita.

As estruturas de repetição podem ser de dois tipos:

- com teste no início, quando a condição de permanência no *looping* é verificada antes da entrada do *looping*;
- com teste no final, quando entra-se no *looping* uma vez e só, então, é verificado se a condição de permanência no *looping* é satisfeita.

Uma forma de estrutura de repetição com teste no início é o comando *Para*, que tem a vantagem de já definir o valor inicial do contador, a condição de permanência no *looping* e o incremento desse contador em uma única linha.

O comando *Para* segue a seguinte estrutura:

Para <contador> de <valor inicial> até <valor final> passo <incremento> faça

Ou seja, *Para j de 1 até 3 passo 1 faça* representa uma estrutura de repetição com teste no início, em que o contador é a variável *j*, que inicia o *looping* com valor 1. Termina-

se o *looping* quando a variável *j* passa a ter valor maior que 3. Executa-se, ainda, o *looping* para *j*=3: seria como se a condição de permanência no *looping* fosse *j*≤3. Veja que *j* é aumentado de uma unidade a cada execução do *looping*.

2. Resolução da questão

Uma ferramenta fundamental para se compreender a lógica de um programa é a simulação, principalmente quando esse programa apresenta estruturas de repetição. Na simulação, é útil trabalharmos com tabelas que contenham uma coluna para cada variável. Para resolvermos a questão, simularemos o trecho de programa dado no enunciado usando uma matriz. Começamos o trecho do programa com:

i<-2

Ou seja, a variável i recebe o valor 2 e o conteúdo dessa variável não é alternado ao longo do programa.

Temos, então, a seguinte estrutura de repetição:

Para j de 1 até 3 passo 1 faça
M[i,j]<-95
M[i,j+1]<-89
M[i,j+2]<-83
M[i,j+3]<-77
Fimpara

Assim, a variável j passa a armazenar o valor 1 nessa primeira execução do *looping.* Como 1≤3, entramos no *looping* e executamos as instruções:

M[2,1]<-95
M[2,2]<-89
M[2,3]<-83
M[2,4]<-77

Esses valores são armazenados na segunda linha da matriz. Ou seja, ficamos com:

95	89	83	77

Ao terminarmos essa primeira execução do *looping*, o contador *j* sofre incremento de 1, ou seja, passa a valer 2.

Temos, ainda, 2≤3 e partimos para a segunda execução do *looping*. Nesse caso:

M[2,2]<-95
M[2,3]<-89
M[2,4]<-83
M[2,5]<-77

Na matriz, os valores anteriores são sobrescritos e temos o seguinte:

95	95	89	83	77

Ao terminarmos essa segunda execução do *looping*, o contador *j* sofre novamente incremento de 1 e passa a valer 3.

Temos, ainda, 3≤3 e partimos para uma terceira execução do *looping*. Nesse caso:

M[2,3]<-95
M[2,4]<-89
M[2,5]<-83
M[2,6]<-77

Na matriz, os valores anteriores são novamente sobrescritos e ficamos com:

95	95	95	89	83	77

Ao terminarmos essa terceira execução do *looping*, o contador *j* sofre incremento de uma unidade e passa a armazenar o número 4. Como, nesse ponto, a condição j≤3 deixa de ser satisfeita, saímos do *looping*.

Note que a primeira linha da matriz permanece vazia.

Alternativa correta: A.

3. Indicações bibliográficas

- BORATTI, I.; OLIVEIRA, A. *Introdução à programação: algoritmos*. Florianópolis: Visual Books, 2004.
- CARBONI, I. *Lógica de programação*. São Paulo: Thomson, 2003.
- MANZANO, J. A. N. G.; OLIVEIRA, J. F. *Algoritmos: lógica para desenvolvimento de programação*. 4. ed. São Paulo: Erica, 2004.

Questão 9

Questão 9.[9]

Considere que um sistema supervisório esteja conectado a um controlador lógico programável (CLP) de forma a receber informações provenientes do processo. A comunicação entre o sistema supervisório e o CLP é realizada por meio de uma rede industrial do tipo mestre-escravo.

Na situação descrita acima, avalie as asserções e a relação proposta entre elas.

I. O sistema supervisório deve ser configurado como o escravo da rede de comunicação.

PORQUE

II. O CLP fornece as informações do processo ao sistema supervisório, uma vez que o CLP recebe as informações das entradas e determina o estado das saídas.

A respeito dessas asserções, assinale a opção correta.

A. As asserções I e II são proposições verdadeiras, e a asserção II justifica a I.

B. As asserções I e II são proposições verdadeiras, e a asserção II não justifica a I.

C. A asserção I é uma proposição verdadeira, e a II é uma proposição falsa.

D. A asserção I é uma proposição falsa, e a II é uma proposição verdadeira.

E. As asserções I e II são proposições falsas.

1. Introdução teórica

1.1. Controladores Lógicos e Programáveis (CLPs)

Os Controladores Lógicos e Programáveis (CLPs ou, no Inglês, PLCs, de *Programmable Logic Controllers*) são dispositivos utilizados na indústria para a automação.

Antes da introdução dos CLPs na indústria, a automação de processos industriais era feita principalmente pela lógica baseada em relés (ALVES, 2010). Contudo, a utilização de quadros de relés apresenta uma série de problemas:

- alto custo de manutenção e instalação da fiação;
- ocupação de elevado espaço físico (um quadro de relês tende a ocupar um espaço físico grande, especialmente se a lógica de controle for complexa);
- elevada produção de interferência eletromagnética (principalmente devido ao faiscamento nos contatos elétricos);
- alto consumo de energia elétrica; e

[9]Questão 28 – Enade 2014.

- necessidade de interrupção do processo industrial no caso de manutenção.

Para superar esses problemas, surgiu a ideia de um dispositivo que tivesse as seguintes características (SILVEIRA e SANTOS, 2008):

- facilidade de diagnóstico;
- ocupação um espaço físico reduzido;
- facilidade de reprogramação e manutenção sem a necessidade de interrupção (programação online);
- baixo consumo de energia;
- elevada confiabilidade;
- baixa interferência eletromagnética, bem como robustez a esse tipo de interferência (típica de ambientes industriais);
- flexibilidade de expansão e comunicação com outros equipamentos.

Os CLPs foram projetados para satisfazer a essas características. Os primeiros CLPs surgiram no final da década de 1960 e início da década de 1970. Na figura 1, há um diagrama de blocos básico de um CLP.

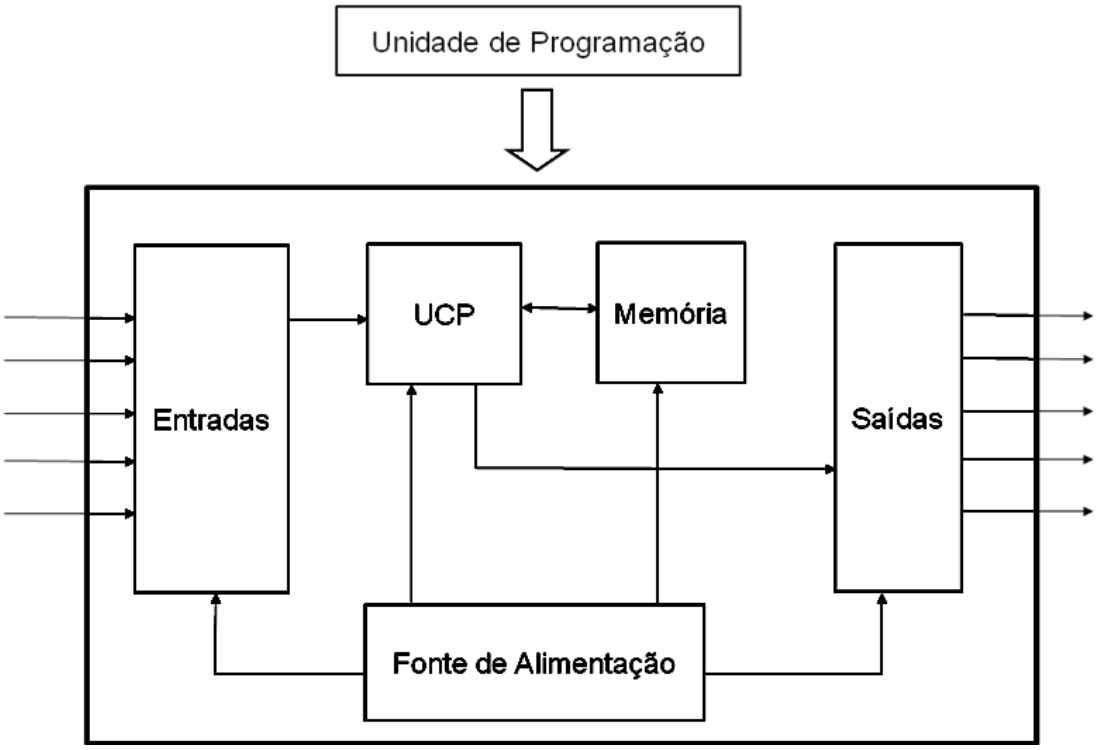


Figura 1. Diagrama básico de um CLP.
Fonte. ALVES, 2010 (com adaptações).

Observa-se que o CLP é composto de entradas (também chamadas "variáveis de entrada"), correspondentes aos sinais externos recebidos pelo CLP, que podem ser tanto gerados por sensores quanto por sinais de controle de um operador (SILVEIRA e SANTOS,

2008). As saídas (ou variáveis de saída) correspondem aos sinais de atuação do CLP para os equipamentos que estão sendo controlados.

Os CLPs também podem ser empregados para sinalização. Internamente, um programa é executado pelo CLP, que utiliza uma UCP e memórias para armazenar as instruções e os parâmetros de execução.

1.2. Modelo CIM (*Computer Integrated Manufacturing*) e redes industriais

Do ponto de vista macroscópico, o modelo CIM (*Computer Integrated Manufacturing*) considera quatros níveis hierárquicos para a integração dos sistemas em uma empresa, conforme segue (SENAI-SP, 2015).

- **Nível de gestão**: é o nível mais elevado, que envolve as atividades administrativas do negócio (vendas e estoques, por exemplo). Nesse nível, as redes integram informações vindas de servidores, *notebooks* e *workstations* por meio de LANs (*Local Area Networks*) ou da Internet, no caso de distâncias longas.
- **Nível de controle**: é o nível em que são feitos os projetos, a programação da produção e o controle de qualidade.
- **Nível de campo e processo**: é o nível em que dispositivos, como CLPs, controladores, transmissores e IHMsP, estão se comunicando e controlando os equipamentos diretamente utilizados na fábrica. Nesse nível, são utilizadas redes industriais.
- **Nível de instrumentação**: é o nível mais próximo do processo, composto de sensores e atuadores. Nesse nível, também são utilizadas redes industriais.

Existem diversas classes de redes industriais, com destaque para os seguintes tipos: *sensor bus*, *device bus*, *field bus* e *data bus*. Cada um desses tipos é mais adequado a um tipo específico de situação. Por exemplo, a redes *sensor bus* são mais empregadas no nível de instrumentação.

Frequentemente, as redes industriais incluem, além de diversos equipamentos, CLPs e sistemas supervisórios. Os CLPs estão diretamente envolvidos no controle dos equipamentos. Já os sistemas supervisórios, normalmente, integram as informações vindas dos diferentes equipamentos de uma planta industrial, o que facilita atividades de monitoramento e manutenção.

2. Análise das asserções

I – Asserção falsa.

JUSTIFICATIVA. O sistema supervisório não deve ser "escravo" da rede. Na arquitetura "mestre-escravo", o dispositivo "mestre" é responsável por enviar comandos aos demais dispositivos "escravos". Os "escravos" costumam ser os dispositivos que estão sendo controlados.

II – Asserção verdadeira.

JUSTIFICATIVA. Como o CLP é responsável por controlar diversos dispositivos, ele tem informações de entrada sobre diversos equipamentos (como os sensores). Ele também tem informações de saída, relativas às ações necessárias para o controle dos equipamentos. Logo, o CLP pode fornecer as informações necessárias para que o sistema supervisório monitore e armazene as informações relativas ao funcionamento do sistema de automação.

Alternativa correta: D.

3. Indicações bibliográficas

- ALVES, J. L. L. *Instrumentação, controle e automação de processos.* 2. ed. Rio de Janeiro: LTC, 2010.
- SENAI-SP. *Integração de sistemas eletrônicos.* São Paulo: SENAI, 2015.
- SILVEIRA, P. R. e SANTOS, W. E. *Automação e controle discreto*. 9 ed. São Paulo: Érica, 2008.
- WILAMOWSKI B. M., IRWIN J. D. *Industrial communication systems*. Boca Raton: CRC Press, 2011.

ÍNDICE REMISSIVO