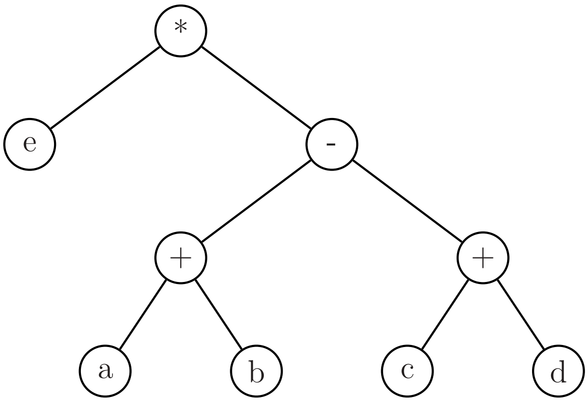
1. Quais das seguintes expressões aritméticas corresponde diretamente à árvore dada no diagrama da figura abaixo.



   (a) $e * (a - c)$

   (b) $e * a - e * c$

   (c) $e * (a + b - c + d)$

   (d) $e * ((a + b) - (c + d))$

   (e) $e * (a + b) - e * (c + d)$

   Em cada subárvore deste tipo de representação de expressões aritméticas, o operador está na raiz, o operando esquerdo no filho (subárvore) esquerdo, e o operando direito no filho direito. Portanto, o percurso em-ordem (in-order), reproduz as operações com as precedências corretas. O percurso em-ordem visita, recursivamente, a subárvore esquerda, a raiz, e a subárvore direita.

   Portanto, a alternativa correta é (d).

2. Em relação às expressões abaixo, quais das seguintes afirmações é verdadeira?

   (i) $\lfloor x \rfloor = \lceil x \rceil$ se e somente se x é um inteiro

   (ii) $\lfloor x + 1 \rfloor = \lceil x \rceil$ se e somente se x não é um inteiro

   (iii) $\lfloor x \rfloor \lceil y \rceil = \lceil x \rceil \lfloor y \rfloor$ para todo x, y

   (iv) $-\lfloor x \rfloor = \lceil -x \rceil$ para todo x

   (a) somente (iv)

   (b) somente (i), (iv)

   (c) somente (i), (ii), (iii)

   (d) somente (i), (ii), (iv)

   (e) (i), (ii), (iii), (iv)

A alternativa (i) está correta, pois os inteiros não têm casas decimais e, portanto, mantêm o mesmo valor depois dos arredondamentos para baixo e para cima.

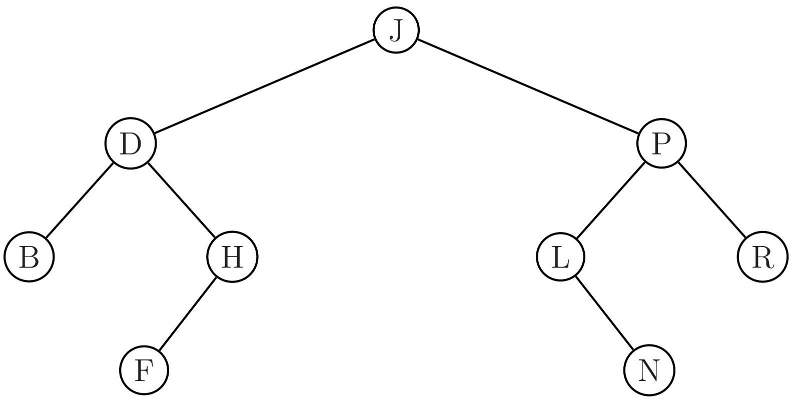
A alternativa (ii) está correta, pois a diferença entre quaisquer dois reais com casas decimais diferentes de zero arredondados, respectivamente, para baixo e para cima, é exatamente 1.

A alternativa (iii) está errada como pode ser constatado pelo contra-exemplo onde $x = 3.5$ e $y = 4.5$, pois $3 \times 5 \neq 4 \times 4$

A alternativa (iv) está correta, pois os números negativos são menores quando os seus valores absolutos são maiores.

Portanto, a alternativa correta é (d).

3. A árvore binária abaixo pode ser usada para guardar uma lista ordenada de modo que um percurso em-ordem (in-order) na árvore gere a lista ordenada. Onde deve ser inserida uma nova entrada K de forma a preservar a ordem alfabética?



   (a) Como filho esquerdo de L
   (b) Como filho esquerdo de N
   (c) Como filho direito de H
   (d) Como filho direito de R
   (e) Como filho direito de F

Para que percurso em-ordem sempre gere uma lista ordenada dos valores dos nós de uma árvore binária, é necessário que a raiz de uma subárvore qualquer seja maior que o seu filho esquerdo e menor que o seu filho direito. Portanto, a entrada K deve ser inserida na subárvore direita de J, na subárvore esquerda de P, e na subárvore esquerda de L, ou seja como filho esquerdo de L.

Portanto, a alternativa correta é (a).

4. Considere um sistema computacional no qual processos podem requerer e liberar um ou mais recursos. Uma vez que um recurso é destinado a um processo, o processo tem uso exclusivo deste recurso até que ele o libera. Se um processo requerer um recurso que está em uso, este processo entra em uma fila para este recurso, esperando até que o recurso esteja disponível. Qual das seguintes opções não resolverá efetivamente o problema de *deadlock*?

   (a) Dar prioridades aos processos e ordenar as filas de espera pela prioridade dos processos.
   (b) Exigir que cada processo requeira todas os seus recursos logo de início, recomeçando o processo caso estes recursos não estejam todos disponíveis.
   (c) Numerar os recursos e exigir que os processos requeiram os recursos em ordem crescente desta numeração.
   (d) Interromper os processos depois de um tempo aleatório e reiniciá-los.
   (e) Fazer o sistema operacional monitorar as filas de espera e recomeçar processos para interromper *deadlocks*.

   A alternativa (a) não resolve o problema, pois além de não haver garantia de execução ordenada dos processos, não se sabe em que ponto de seus códigos um determinado recurso é requisitado. Assim, um processo pode requerer e obter um recurso A enquanto outro requer e obtém um recurso B. Após estes dois eventos, o primeiro processo tenta obter o recurso B enquanto o segundo tenta obter o recurso A, causando o *deadlock*.

   A alternativa (b) está correta pois cada processo tem a garantia de obter todos os recursos que requisitar e poder trabalhar até o seu fim e liberá-los.

   A alternativa (c) está correta, pois um processo com um recurso obtido jamais poderá tentar obter um recurso já obtido por outro processo.

   A alternativa (d) está correta, pois elimina os eventuais *deadlocks* em andamento.

   A alternativa (e) está correta o problema, pois interrompe e recomeça os processos travados.

   Portanto, a alternativa correta é (a).

5. Em uma linguagem na qual as operações têm associatividade da direita para a esquerda ao invés de da esquerda para a direita (isto é, $a+b+c = a+(b+c)$), qual é o valor da expressão: $7 - (16/(3+1)*2) - 4$

   (a) -1
   (b) 1
   (c) 3
   (d) 7
   (e) 9

A expressão é executada assim, uma operação por vez:

(i) $7 - (16/(3 + 1) * 2) - 4$
(ii) $7 - (16/4 * 2) - 4$
(iii) $7 - (16/8) - 4$
(iv) $7 - 2 - 4$
(v) $7 - (-2)$
(vi) 9

Portanto, a alternativa correta é (e).

6. Qual é a melhor aproximação da razão entre o número de nós não-terminais (não-folhas) e o número total de nós de uma árvore k-ária completa com profundidade n?

(a) $1/n$
(b) $n - 1/n$
(c) $1/k$
(d) $k - 1/k$
(e) $\log_k(1/n)$

Na profundidade zero, há apenas a raiz (1 nó). Na profundidade 1 há os k filhos da raiz (k nós). Na profundidade 2, cada um dos k nós produz k filhos, portanto, há k^2 nós. Na profundidade 3, há k^3 nós. Na profundidade n, há (indução) k^n nós; este é o número de folhas da árvore completa.

O número de nós de cada nível forma uma progressão geométrica de razão k, cujo primeiro termo é k^0. O número de nós internos (não-folhas) é a soma dos nós até a profundidade $n - 1$, ou seja, a soma de n termos da p.g.; o número total de nós é a soma de $n + 1$ termos da p.g.:

$$n_1 = \frac{k^n - 1}{k - 1} \qquad n_2 = \frac{k^{n+1} - 1}{k - 1}$$

A razão entre estes dois valores é:

$$\frac{n_1}{n_2} = \frac{k^n - 1}{k - 1} \times \frac{k - 1}{k^{n+1} - 1} = \frac{k^n - 1}{k^{n+1} - 1} \approx \frac{1}{k}$$

Portanto, a alternativa correta é (c).

7. Assuma que uma lista é construída de elementos com dois campos, *val* e *prox*, sendo que o campo *prox* de cada elemento é usado para apontar o próximo na lista. Se os elementos da lista são descritos em C por:

```
typedef struct node *NodePtr;
struct node {
  int val;
  NodePtr prox;
};
```

qual das seguintes é uma implementação correta em C da operação *insira p depois de q*, onde *q* aponta um elemento da lista e *p* para um elemento a ser inserido?

(a) `q->prox = p->prox; p->prox = q;`

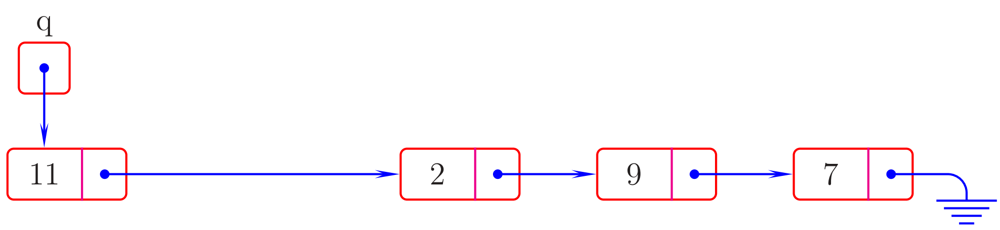
(b) `p->prox = q->prox; q->prox = p;`

(c) `q->prox = p->prox; p->prox = q->prox;`

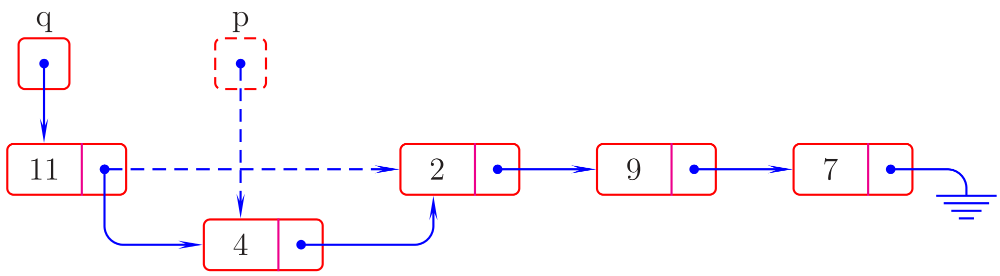
(d) `p->prox = q;       q->prox = p->prox;`

(e) `q->prox = p;       p->prox = q->prox;`

A inserção de um novo nó pode ser mostrada através das figuras abaixo:



(a) Antes da Inserção



(b) Após a Inserção

Portanto, a alternativa correta é (b)

8. Assuma que a seguinte tabela é a seqüência de páginas referenciadas por um programa a ser executado em uma memória com duas páginas com o sistema de paginação por demanda.

Instante	Página referenciada
1	1
2	2
3	1
4	3
5	4
6	pare

Que páginas estão na memória no instante 6 se o programa está executando sob a regra *LFU* (*least frequently used*) de substituição de páginas?

(a) 1 e 3
(b) 1 e 4
(c) 2 e 3
(d) 2 e 4
(e) 3 e 4

Esta estratégia conta as referências de cada página e mantém na memória as páginas com os contadores mais altos. Até o instante 2, há duas páginas (1 e 2) na memória, ambas com uma referência. No instante 3, a página 1 é novamente referenciada, o que aumenta seu contador para 2. Daí para a frente, todas as referências são feitas a novas páginas e, portanto, a página 3 substitui a 2 e, em seguida, a página 4 substitui a 3. No instante 6, as páginas 1 e 4 estão na memória.

Portanto, a alternativa correta é (b).

9. Assuma que os valores das expressões booleanas "$a > b$" e "$b > c$" sejam independentes e que, em média, "$a > b$" ocorre em 75% do tempo e "$b > c$" ocorre em 25% do tempo. Se o fragmento de programa abaixo é executado 10000 vezes, quantas vezes se espera que as funções f e g sejam executadas?

```
if a > b
  then v[i] := f(i)
  else if b > c
    then v[i] := g(i)
```

(a) 2500 e 18750
(b) 7500 e 625

(c) 7500 e 1875

(d) 7500 e 2500

(e) 9375 e 625

O primeiro *if* sempre é executado e, portanto, a probabilidade de executar a função f é 75%, o que significa que ela é executada, em média, $0.75 \times 10000 \; = = 7500$ vezes.

O segundo *if* somente é executado quando o primeiro falha, o que ocorre com probabilidade $1 - 0.75 \; = \; 0.25$. Como as duas condições são independentes, por definição do problema, a função g é executada com probabilidade 0.25×0.25, o que significa que ela é executada, em média, $0.25 \times 0.25 \times 10000 \; = \; 625$ vezes.

Portanto, a alternativa correta é (b).

10. Qual é a condição menos restritiva que deve ser necessariamente verdadeira antes da execução do comando abaixo para que se tenha certeza que após a sua execução x seja igual ao valor de a?

```
if a > b then x := a;
```

(a) $(x = a) \vee (a > b)$

(b) $(a > b) \wedge (x = a)$

(c) $a > b$

(d) $x > b$

(e) $x = a$

A condição (b) satisfaz o enunciado do problema mas é excessiva, já que bastaria a primeira expressão ser verdadeira ($a > b$). A condição (c) é menos restritiva que a (b) e satisfaz o enunciado. A condição (d) não satisfaz, necessariamente o enunciado, pois não garante nada. A condição (e) satisfaz o enunciado, mas exige que x seja igual a a. A condição (a) satisfaz o enunciado e é a menos restritiva de todas, pois permite, sem exigência, que x tenha o valor a ou que a condição do *if* seja verdadeira.

Portanto, a alternativa correta é (a).

11. Na função abaixo, x é passado por referência e y, por valor.

```
function f (var x: integer; y: integer): integer;
  begin
    k := 3;
    m := 5;
    f := x + y;
  end;
```

Qual será o valor de z se a função f for chamada pelo seguinte fragmento de programa:

```
k := 1;
m := 1;
z := f(k,m);
```

(a) 2

(b) 3

(c) 4

(d) 6

(e) 8

O parâmetro y é passado por valor e nem é modificado; portanto, seu valor dentro da função é sempre 1 (valor do argumento m). O parâmetro x é passado por referência e, portanto, é um outro nome para a variável global k. Como k é modificada dentro da função, este é o valor do parâmetro x (3). Portanto, o valor de retorna da função, que é o valor da variável z é $3 + 1 = 4$.

Portanto, a alternativa correta é (c).

12. Uma relação pode ser definida dando-se a lista de pares para os quais ela vale. Quais propriedades que a relação $\mathcal{R}$, definida abaixo sobre $A = \{a, b, c\}$, tem?

$\mathcal{R} = \{(a, a), (a, b), (b, a), (b, b), (c, c)\}$

(a) Nenhuma

(b) Anti-simetria e reflexividade

(c) Anti-simetria e transitividade

(d) Simetria, reflexividade, transitividade

(e) Anti-simetria, reflexividade e transitividade

Para ser simétrica, $a\,\mathcal{R}\,b \Rightarrow b\,\mathcal{R}\,a$, o que é verdade, pois ela vale para os pares com componentes iguais e para (a, b) e (b, a).

Para ser anti-simétrica, $a\,\mathcal{R}\,b$ e $b\,\mathcal{R}\,a \Rightarrow a = b$, o que não é verdade, pois ela vale para os pares com componentes diferentes (a, b) e (b, a).

Para ser reflexiva, $\forall\, a \in A\,,\; a\,\mathcal{R}\,a$, o que é verdade pois ela vale para todos os três pares com componentes iguais.

Para ser transitiva, $a\,\mathcal{R}\,b$ e $b\,\mathcal{R}\,c \Rightarrow a\,\mathcal{R}\,c$, o que é verdade.

Portanto, a alternativa correta é (d).

13. Assuma que um arquivo de dados tem um índice que consiste de n itens (n é grande). Se uma busca binária do índice é usada para encontrar um item, qual é a melhor aproximação da média de acessos ao arquivo para localizar uma entrada específica do índice?

    (a) $(n+1)/2$

    (b) $n(n-1)/2$

    (c) $\log_2(n) - 1$

    (d) $n \log_2(n)$

    (e) $(n+1)/\log_2(n)$

    O algoritmo de busca binária consegue encontrar um item com uma média de $\log_2(n) - 1$ iterações.

    Portanto, a alternativa correta é (c).

14. Uma expressão regular que denota todas as strings de 0's e 1's que têm pelo menos dois 1's consecutivos é

    (a) (0+10)*11(10+0)*

    (b) (0+10)*11(0+1)*

    (c) (0+1)*10*1(0+1)*

    (d) 0*11(0+10)*(0+1)*

    (e) (0+11)*

    A expressão (a) não está correta porque, uma vez gerada uma seqüência 11, não há como gerar outras eventuais seqüências 11. Por exemplo, o string 1111 não é gerado por esta expressão.

    A expressão (b) garante a geração da seqüência 11 seguida de quaisquer quantidades de 0 e 1. Antes da seqüência 11, poderá existir uma combinação qualquer de 0 ou 10, que resolve o problema. O string 01001110 é gerado assim: uma instância de 0 de (0+10)*, uma instância de 10 de (0+10)*, uma instância de 0 de (0+10)*, uma seqüência 11, uma instância de 1 de (0+1)*, uma instância de 0 de (0+1)*.

    A expressão (c) gera, por exemplo, a seqüência 01010, que não satisfaz o enunciado.

    A expressão (d) não gera, por exemplo, o string 1111.

    A expressão (e) gera, por exemplo, um string vazio.

    Portanto, a alternativa correta é (b), correspondente a um autômato finito determinístico, mas a expressão (0+1)*11(0+1)*, correspondente a um autômato finito não-determinístico, é mais clara.

15. Para $n = 6$ e os elementos do vetor a iguais, respectivamente, a $-8, 4, 10, -2, 7, 3$, quais serão os valores dos elementos de a após a execução do trecho de programa abaixo? Assuma que todas as variáveis são do tipo inteiro e que $n > 1$.

```
(1)  for i := 1 to n-1 do
(2)    for j := i+1 to n do
(3)      if a[i] < a[j] then begin
(4)         t := a[j];
(5)         a[j] := a[i];
(6)         a[i] := t
(7)      end
```

(a) 10, 7, 4, 3,-2,-8

(b) 10,-8, 7, 4, 3,-2

(c) 4,-8,-2,10, 3, 7

(d) -2, 3, 4, 7,-8,10

(e) -8,-2, 3, 4, 7,10

As iterações do *for* interno trocam o elemento corrente (em j) com o seu primeiro, segundo, etc., elementos do array (em i), se este for menor. O estado inicial do array e após cada iteração do *for* interno estão mostrados na tabela abaixo:

```
-8  4 10 -2 7 3    10 -8  4 -2 7 3    10 7 -8 -2  4 3    10 7 4 -8 -2  3    10 7 4  3 -8 -2
 4 -8 10 -2 7 3    10  4 -8 -2 7 3    10 7 -2 -8  4 3    10 7 4 -2 -8  3    10 7 4  3 -2 -8
10 -8  4 -2 7 3    10  4 -8 -2 7 3    10 7  4 -8 -2 3    10 7 4  3 -8 -2
10 -8  4 -2 7 3    10  7 -8 -2 4 3    10 7  4 -8 -2 3
10 -8  4 -2 7 3    10  7 -8 -2 4 3
10 -8  4 -2 7 3
```

Portanto, a alternativa correta é (a).

16. Em relação ao trecho de programa da questão (15), qual é o maior número possível de vezes, em termos de n, que o bloco das linhas (4) até (6) é executado?

(a) $n(n-1)/2$

(b) $n \log_2(n)$

(c) n^2

(d) $n(n+1)/2$

(e) Não é possível calcular

Para cada iteração do *for* externo, o *for* interno, cujo valor de início é variável, executa o bloco em questão, $(n-1), (n-2), (n-3) \ldots 1$ vezes; portanto:

$$(n-1) + (n-2) + \cdots 1 \;=\; \frac{(n-1+1)(n-1)}{2} \;=\; \frac{n(n-1)}{2}$$

Portanto, a alternativa correta é (a).

17. Em relação ao trecho de programa da questão (15), qual é o conteúdo de a após a execução, se a ordem das instruções (4) e (5) for invertida de forma que (4), (5), e (6) sejam agora:

```
(4)  a[j] := a[i];
(5)  t := a[j];
(6)  a[i] := t
```

(a) O mesmo que antes para todo a

(b) Independente do conteúdo inicial de a

(c) Diferente do inicial para todo n

(d) Diferente do inicial para alguns valores de n e de a

(e) Imprevisível para alguns valores de n e de a

A alteração faz os dois elementos ficarem iguais ao menor em vez de trocar os seus conteúdos. O resultado pode até não fazer diferença em alguns casos particulares, mas será diferente na maioria dos casos.

Portanto, a alternativa correta é (d).

18. Expressões em uma certa linguagem podem ser descritas na chamada forma de Backus-Naur (BNF) da seguinte maneira:

<E> ::= <T> | <E> op1 <T>
<T> ::= <F> | <T> op2 <F>
<F> ::= <V> | <N>

onde V é uma variável qualquer e N é um número qualquer. Esta sintaxe é mais apropriada quando a ordem de cálculo (associatividade) é:

(a) Da esquerda para a direita sempre

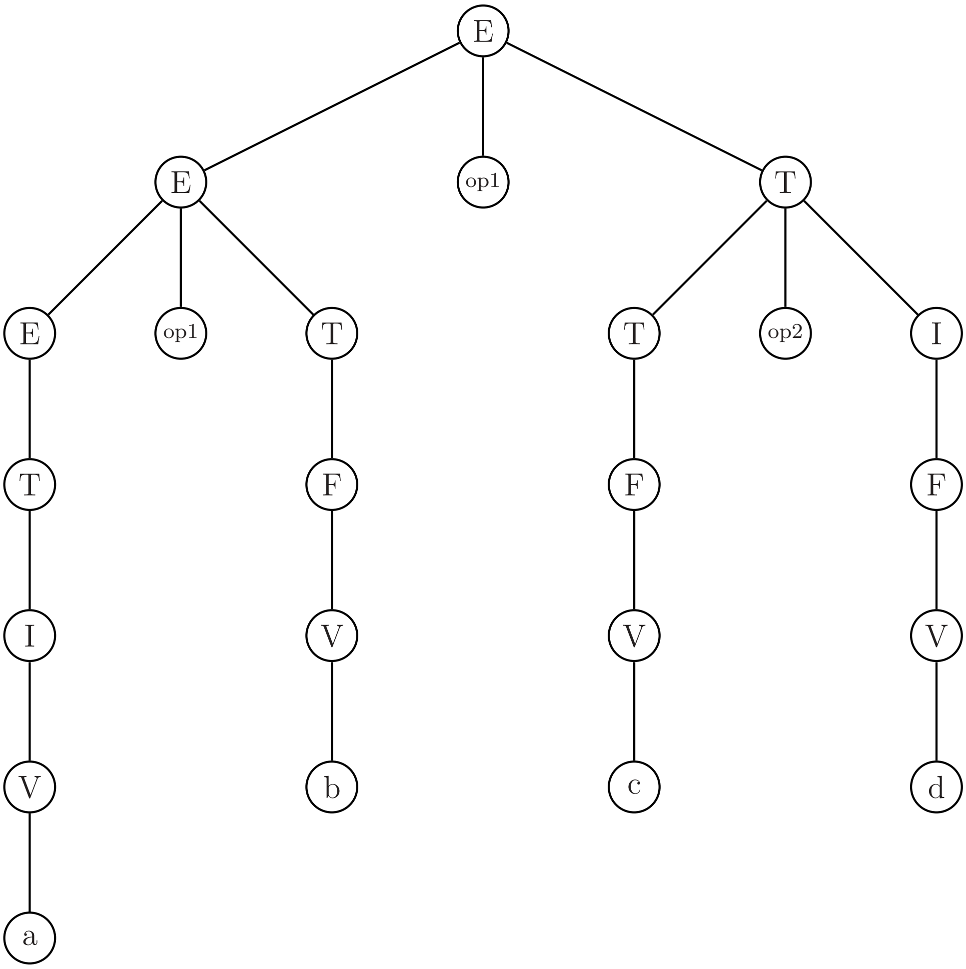
(b) Da esquerda para a direita, mas o operador $op1$ tem precedência sobre $op2$

(c) Da esquerda para a direita, mas o operador $op2$ tem precedência sobre $op1$

(d) Em qualquer ordem, mas $op1$ tem precedência sobre $op2$

(e) Da direita para a esquerda sempre

Uma derivação da expressão $a \quad op1 \quad b \quad op1 \quad c \quad op2 \quad d$ é:



Pode-se ver pelo percurso em-ordem que a associatividade é pela esquerda e $op2$ tem precedência sobre $op1$.

Portanto a resposta correta é (c).

19. Considere as seguintes operações em strings:

- $head(x)$ = primeiro caractere de x se $x \neq \varepsilon$ e indefinido em caso contrário
- $tail(x) = x$ sem o primeiro caractere (ε, se x tiver menos que dois caracteres)
- $join(x, y)$ = string obtida da concatenação de x e y

Se x é uma string não vazia, qual dos seguintes é necessariamente verdade?

(a) $join(head(x),tail(y)) = x$

(b) $head(tail(x)) = tail(head(x))$

(c) $join(head(x),x) = join(x,tail(x))$

(d) $head(tail(x)) = x$

(e) $tail(head(x)) = x$

A alternativa (a) é verdadeira porque qualquer string não vazio é a concatenação de seu primeiro caractere com o resto.

A alternativa (b) afirma que o segundo caractere de um string (o primeiro do resto) é vazio (resto de um string que tem apenas um caractere), o que só é verdadeiro para strings com um caractere.

A alternativa (c) afirma que duplicar o primeiro caractere de um string torna-o igual à concatenação deste string com o seu resto (sem o primeiro), o que é verdadeiro apenas para strings de dois caracteres idênticos.

A alternativa (d) afirma que o segundo caractere de um string é igual ao string, o que nunca é verdadeiro.

A alternativa (e) afirma que o resto do string (a partir do segundo) é igual ao string, o que só verdadeiro para strings vazios.

Portanto, a alternativa correta é (a).

20. Qual dos seguintes inverte a ordem dos caracteres em uma string com pelo menos dois caracteres se inverso(x) = x quando x é uma string com um único caracter?

   (a) $inverso(x) = join(inverso(tail(x)), head(x))$
   (b) $inverso(x) = join(tail(inverso(x)), head(x))$
   (c) $inverso(x) = join(tail(x), head(x))$
   (d) $inverso(x) = inverso(join(head(x), tail(x)))$
   (e) $inverso(x) = join(head(x), inverso(tail(x)))$

   A alternativa (a) concatena, recursivamente, o inverso do resto do string com o primeiro caractere, o que está correto. Cada passo desta chamada recursiva reduz o tamanho do string até o caso base.

   A alternativa (b) chama recursivamente a função sem reduzir o tamanho do string original; portanto está incorreta.

   A alternativa (c) concatena o resto do string com o primeiro caractere, o que simplesmente move o primeiro caractere para o fim do string, o que está incorreto.

   A alternativa (d) chama recursivamente a função com o parâmetro original, ou seja, sem reduzir o seu tamanho; portanto está incorreta.

   A alternativa (e) concatena o primeiro caractere com o inverso do resto. Embora a chamada recursiva não entre em loop, o procedimento não produz o resultado desejado.

   Portanto, a alternativa correta é (a).

21. Se todas as variáveis são do tipo inteiro e se $a = b \pmod{m}$ e $x = y \pmod{m}$, quais das afirmativas abaixo são verdadeiras?

(i) $a + x = b + y \pmod{m}$

(ii) $ax = by \pmod{m}$

(iii) $a/n = b/n \pmod{m}$ para todo $n \neq 0$

(a) Somente (ii)

(b) Somente (iii)

(c) Somente (i) e (ii)

(d) Somente (i) e (iii)

(e) (i), (ii), (iii)

Por definição de divisão, o dividendo é igual ao divisor multiplicado pelo quociente mais o resto (todos inteiros). Portanto:

$$a = b \pmod{m} \quad \Rightarrow \quad a = q_1 m + b \tag{1}$$

$$x = y \pmod{m} \quad \Rightarrow \quad x = q_2 m + y \tag{2}$$

Somando as equações (1) e (2):

$$a + x = (q_1 + q_2)m + (b + y) \quad \Rightarrow \quad a + x = b + y \pmod{m}$$

Multiplicando as equações (1) e (2):

$$ax = (q_1 q_2 m + q_1 y + q2b)m + by \quad \Rightarrow \quad ax = by \pmod{m}$$

Dividindo a equação (1) por $n \neq 0$:

$$\frac{a}{n} = \frac{q_1}{n} m + \frac{b}{n} \tag{3}$$

A equação (3) só implica em $a/n = b/n \pmod{m}$ se q_1 e b forem múltiplos de n, o que não é sempre verdadeiro.

Portanto, apenas as afirmativas (i) e (ii) estão corretas e a alternativa correta é (c).