Fundação CECIERJ - Vice Presidência de Educação Superior a Distância

Curso de Tecnologia em Sistemas de Computação
Disciplina: Projeto e Desenvolvimento de Algoritmos
AP3 2° semestre de 2010.

Nome –

Assinatura –

Observações:

1. Prova sem consulta e sem uso de máquina de calcular.
2. Use caneta para preencher o seu nome e assinar nas folhas de questões e nas folhas de respostas.
3. Você pode usar lápis para responder as questões.
4. Ao final da prova devolva as folhas de questões e as de respostas.
5. **Todas as respostas devem ser transcritas no local apropriado, no cartão de respostas a seguir.**

Questão					
1	A	B	■	D	E
2	A	■	C	D	E
3	A	B	C	■	E
4	A	B	C	■	E
5	A	B	C	D	■
6	■	B	C	D	E
7	■	B	C	D	E
8	A	B	■	D	E
9	A	B	C	■	E
10	A	B	C	■	E

dtkyLG99tig7eyrRFLhUgHwBt4kbTJN7tuZhBLNxa2z8Y5pEys

1ª questão (valor 1.0)

O algoritmo a seguir calcula o número de voltas que a roda de uma bicicleta dará para percorrer uma distância fornecida pelo usuário. O usuário fornece o raio da roda (em centímetros) e a distância a percorrer (em quilômetros).

```
início
    PI ← 3.14
    imprima 'distância a percorrer (km)? '
    leia distancia
    distancia ← 100000 * distancia
    imprima "Raio da roda (cm)? "
    leia raio
    contaVoltas ← 0
    enquanto distancia > 0 faça
        contaVoltas ← contaVoltas + 1
        distancia ← distancia - 2 * PI * raio
    fim enquanto
    imprima contaVoltas
fim
```

Se os valores fornecidos pelo usuário forem 100 e 33, nessa ordem, a saída do programa será:

A) 1
B) **22587**
C) **48254**
D) **78522**
E) Nenhuma das respostas anteriores

2ª questão (valor 1.0)

Na análise do algoritmo a seguir, considere a existência da função **pow(entradas: x, y)** que retorna um número de ponto flutuante correspondente ao valor de x^y (x elevado a y)

```
variáveis públicas
    NUM_TERMOS

função calculoFacil(entradas: num)
início
    soma ← 0.0
    mult ← 1
    j ← 1
    para i ← 1 até NUM_TERMOS faça
        soma ← soma + pow(num, j)/j*mult
        mult ← -mult
        j ← j + 2
    próximo i
    resultado ← soma
fim

início
    NUM_TERMOS ← 3
    imprima 'Entre com um número de ponto flutuante: '
    leia num
    imprima calculoFacil(num)
fim
```

Se o valor de entrada fornecido pelo usuário for o número 1.0, a saída impressa pelo algoritmo será:

A) **0.00000**
B) **0.86667**
C) **1.00000**
D) **1.53333**
E) Nenhuma das respostas anteriores

3ª questão (valor 1.0)

Observe o algoritmo a seguir:

```
início
    resultado ← 1
    acabou ← falso
    enquanto não acabou faça
        imprima 'Dê-me um número: '
        leia n1
        se n1 = 0 então
            acabou ← verdadeiro
        senão
            resultado ← resultado * n1
        fim se
    fim enquanto

    imprima resultado
fim
```

Se os valores fornecidos pelo usuário ao algoritmo forem **5**, **4**, **3**, **2**, **1** e **0**, nessa ordem, o valor impresso pelo algoritmo será:

A) **0**
B) **1**
C) **15**
D) **120**
E) Nenhuma das respostas anteriores

4ª questão (valor 1.0)

O que será impresso pelo algoritmo a seguir?

```
procedimento foo(entradas: num)
início
    num ← 4
    x ← 3
fim

início
    x ← 5
    foo(x)
    imprima x
fim
```

A) **0**
B) **3**
C) **4**
D) **5**
E) Nenhuma das respostas anteriores

5ª questão (valor 1.0)

Na análise do algoritmo a seguir, considere a existência das funções **charAt()** e **len()** cuja documentação é mostrada a seguir:

charAt(string, indice)

Retorna o caractere na posição especificada pelo índice. O índice pode variar de 1 até o tamanho da string. O primeiro caractere da sequência tem o índice 1, o seguinte o índice 2 e assim por diante.

Exemplo:

```
imprima charAt('CEDERJ', 2)
```

imprimiria o caractere **'E'**

len(string)

Retorna o tamanho da string. O tamanho é igual ao número de caracteres na string.

Exemplo:

```
imprima len('CEDERJ')
```

imprimiria o inteiro **6**

```
início
    nome ← 'Barack Hussein Obama'
    inicioPalavra ← verdadeiro
    para i ← 1 até len(nome) faça
        se inicioPalavra então
            imprima charAt(nome, i)
        fim se
        se nome[i] = ' ' então
            inicioPalavra ← verdadeiro
        senão
            inicioPalavra ← falso
        fim se
    próximo i
fim
```

A saída do algoritmo será:

A) **B**
B) **Barack**
C) **Obama**
D) **Barack Hussein Obama**
E) Nenhuma das respostas anteriores

6ª questão (valor 1.0)

Observe o algoritmo a seguir.

```
função gcd(entradas: u,v)
início
    se u < v então
        t ← u
    senão
        t ← v
    fim se
    enquanto (u mod t <> 0) OU (v mod t <> 0) faça
        t ← t - 1
    fim enquanto
    resultado ← t
fim

início
    leia x
    leia y
    imprima gcd(x,y)
fim
```

Se os números fornecidos pelo usuário forem **8** e **6**, a saída impressa pelo algoritmo será:

A) 2
B) 4
C) 6
D) 8
E) Nenhuma das respostas anteriores

7ª questão (valor 1.0)

Observe o algoritmo a seguir.

```
variáveis globais: DIM

procedimento fler (saídas: v[ ])
início
    para i ← 1 até DIM faça
        leia v[i]
    próximo i
fim

procedimento fimprimir (entradas: v[ ])
início
    para i ← 1 até DIM faça
        imprima v[i]
    próximo i
fim
```

```
procedimento pm (entradas: p[ ],q[ ]
                 saídas: r[ ])
início
    para i ← 1 até 2*DIM - 1 faça
        r[i] ← 0.0
    próximo i
    para i ← 1 até DIM faça
        para j ← 1 até DIM faça
            r[i+j-1] ← r[i+j-1] + p[i] * q[j]
        próximo j
    próximo i
fim

início
    DIM ← 3
    fler (p)
    fler (q)
    pm(p,q,r)
    fimprimir(r)
fim
```

Considere que os valores digitados pelo usuário foram 1, 2, 3, 2, 2, 2, nessa ordem.
A saída impressa pelo programa será:

A) `2, 6, 12, 10, 6`
B) `6, 10, 12, 6, 2`
C) `1, 2, 3, 2, 2, 2`
D) `2, 2, 2, 1, 2, 3`
E) Nenhuma das respostas anteriores

8ª questão (valor 1.0)

Observe o algoritmo a seguir:

```
procedimento lev (entradas: t
                  saídas: v[ ])
início
    para i ← 1 até t faça
        leia v[i]
    próximo i
fim

função qp(entradas v[ ], t)
início
    c ← 0
    para i ← 1 até t faça
        se v[i] mod 2 = 0 então
            c ← c + 1
        fim se
    próximo i
    resultado ← c
fim

procedimento imp (entradas: v[ ], t, p)
início
    para i ← p até t faça
        imprima v[i]
    próximo i
fim
```

```
início
    DIM ← 6
    lev(v, DIM)
    p ← qp(v, DIM)
    se p < DIM então
        imp(v, DIM, p)
    fim se
fim
```

Se os valores 1, 2, 3, 4, 5, 6 forem fornecidos nessa ordem ao algoritmo, a saída será:

A) 1 2 3 4 5 6
B) 6 5 4 3 2 1
C) 3 4 5 6
D) 1 2 3 4
E) Nenhuma das respostas anteriores

9ª questão (valor 1.0)

Considere que, em PETEQS, existe uma tabela do tipo ASCII que atribui um número inteiro para cada caractere. Considere ainda que estão disponíveis as seguintes funções:

```
ordem(car) { retorna a posição do caractere car na tabela de caracteres do
             computador }

caractere(num) { retorna o caractere cuja posição na tabela de caracteres do
                 computador seja num }

leCadeia(frase) { lê um conjunto de caracteres do teclado e os armazena no vetor
                  frase }

compCadeia(frase) {retorna quantos caracteres estão armazenados no vetor frase}

charAt(frase, indice) {retorna o caractere na posição indice}
```

Os caracteres alfabéticos ocupam posições contíguas na tabela, isto é,
ordem('B') - ordem('A') = 1

e portanto,

caractere(ordem('A') + 1) = 'B'

Usando estas funções, um aluno de PDA escreveu o seguinte algoritmo:

```
início
    leCadeia(frase)
    c ← 1
    para i ← 1 até compCadeia(frase) faça
       c ← c * (ordem(charAt(frase, i) - ordem('A'))
    próximo i
    imprima c
fim
```

Marque a opção que mostra o que será impresso pelo algoritmo caso o usuário forneça a frase **BCDEF** como entrada.

A) 1
B) 30
C) 60
D) 120
E) Nenhuma das respostas anteriores

10ª questão (valor 1.0)

Considere que, em PETEQS, existe uma tabela do tipo ASCII que atribui um número inteiro para cada caractere. Considere ainda que estão disponíveis as seguintes funções:

```
leCadeia(frase) { lê um conjunto de caracteres do teclado e os armazena no vetor
                  frase }

compCadeia(frase) {retorna quantos caracteres estão armazenados no vetor frase}

charAt(frase, indice) {retorna o caractere na posição indice}
```

Usando estas funções, um aluno de PDA escreveu o seguinte algoritmo:

```
função Conta(entradas v[ ])
início
    c ← 0
    para i ← 1 até compCadeia(v) faça
        se (charAt(v, i) = 'e') OU (charAt(v, i) = 'E') então
            c ← c + 1
        senão
            se (charAt(v, i) = 'v') OU (charAt(v, i) = 'V') então
                c ← c + 3
            fim se
        fim se
    próximo i
    resultado ← c
fim

início
    leCadeia(frase)
    p ← Conta(frase)
    imprima p
fim
```

Marque a opção que mostra o que será impresso pelo algoritmo caso seja digitado o seguinte conjunto de caracteres:

EDDDEVVEE

A) 2
B) 4
C) 8
D) 10
E) Nenhuma das respostas anteriores