Dredd - Juiz Online

Minutos Restantes: 747

Usuário: Wesley de Jesus Santos

Notas:
Q1: ?
Q2: ?
Q3: ?
Q4: ?
Q5: ?
Q6: ?
Q7: ?
Q8: ?
Q9: ?
Q10: ?
Total: 0

IAlg EAD - Exercícios de Recursão - 2019/1 - 22/04 à 28/04

Prova Aberta Até: 26/04/2019 23:59:59

Número Máximo de Tentativas: 6

Atenuação da Nota por Tentativa: 0%

Instruções para a prova: Lista de exercícios de Recursão para todas turmas de IAlg. Pode ser acessada de casa.

Contabilização de frequência

Os exercícios dessa semana que serão utilizados para a contabilização de frequência são: 5 e 7.

Para que a frequência seja contabilizada o aluno deve, ao menos, ter tentado resolver o exercício, não sendo necessário que tenha obtido nota máxima. **Não** serão aceitos para contabilização de frequência tentativas "vazias" - envio de código que compila, mas não tenta resolver o problema. Além disso, **poderão ser utilizadas ferramentas para detecção de plágio e, em caso de plágio, todos os envolvidos ficarão com falta**.

Questão 1: Recursividade - Número áureo

O *número áureo*, frequentemente denotado pela letra grega ϕ (phi) é um número real irracional que ocorre espontaneamente na natureza e é frequentemente usado nas artes por estar relacionado à nossa percepção de beleza.

O *número áureo* pode ser calculado pela recorrência ***ϕ = 1 + 1/ϕ***.

Por ser uma recorrência infinita, ela precisa ser limitada para ser usada na recursividade da Computação. Podemos definir o valor aproximado de ϕ em função do número de termos usados no cálculo, assim:

$$phi(n) = \begin{cases} \bullet 1, & se\ n = 1; \\ \bullet 1 + \frac{1}{phi(n-1)}, & se\ n > 1. \end{cases}$$

Faça um programa que tem uma função que calcula uma aproximação do *número áureo*, usando recursão.

O *número áureo,* deve ser do tipo *ponto flutuante de precisão dupla* (`double`) para possibilitar a precisão necessária nos cálculos. As operações de leitura e escrita devem ser realizadas na função principal.

Entradas:

1. O número de termos para o cálculo da aproximação do *número áureo*.

Saídas:

1. O valor aproximado do *número áureo*.

Exemplo de Entrada:

```
3
```

Exemplo de Saída:

```
1.5
```

Peso: 1

Nova Resposta:

Selecione o arquivo com o código fonte do programa que resolve o problema para enviá-lo.

Escolher Arquivo nenhum arquivo selecionado Enviar Resposta

Questão 2: Recursividade - Somas sucessivas.

(BACKES,2012) A multiplicação de dois números naturais pode ser feita através de somas sucessivas (por exemplo, 2*3=2+2+2). Crie uma **função recursiva** que calcule a multiplicação por somas sucessivas de dois números naturais.

Entradas:

1. Dois números naturais.

Saídas:

1. Resultado da multiplicação dos dois números.

Exemplo de Entrada:

```
3 5
```

Exemplo de Saída:

```
15
```

Peso: 1

Nova Resposta:

Selecione o arquivo com o código fonte do programa que resolve o problema para enviá-lo.

Escolher Arquivo nenhum arquivo selecionado Enviar Resposta

Questão 3: Recursividade – Impressão por recursão

Crie um subprograma **recursivo** que receba um número inteiro N e imprima todos os números naturais de 0 até N em ordem crescente.

Entradas:

1. Um número inteiro

Saídas:

1. Uma sequência de números naturais de 0 até N

Exemplo de Entradas:

```
15
```

Exemplo de Saída:

```
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
```

Peso: 1

Nova Resposta:

Selecione o arquivo com o código fonte do programa que resolve o problema para enviá-lo.

Escolher Arquivo nenhum arquivo selecionado Enviar Resposta

Questão 4: Recursividade – Sequência de Ricci

A sequência de Ricci é uma sequência bastante semelhante à de Fibonacci, difererindo desta apenas pelo fato que os dois primeiros termos da sequência (F(0) e F(1)) devem ser definidos pelo usuário.

Sabendo-se que a sequência de Fibonacci é definida por:

- F(0) = 0
- F(1) = 1
- F(n) = F(n - 1) + F(n - 2) , n ≥ 2

Crie um algoritmo que imprima os n primeiros termos da sequência de Ricci, utilizando um **subprograma** que retorna o n-ésimo termo da referida série.

Entrada:

1. Os valores iniciais da série de Ricci (F(0) e F(1));
2. Os número de termos dessa sequência a serem impressos.

Saída:

1. Os n termos dessa sequência.

Exemplo de entrada:

```
5 8
6
```

Exemplo de saída:

```
5 8 13 21 34 55
```

Peso: 1

Nova Resposta:

Selecione o arquivo com o código fonte do programa que resolve o problema para enviá-lo.

Escolher Arquivo nenhum arquivo selecionado Enviar Resposta

Questão 5: Recursividade - Coeficientes Binomiais

Denote por C(N,K) o número de possibilidades de se escolherem K dentre N elementos de um conjunto. Este valor é dado pela seguinte relação de recorrência:

- C(N, 0) = 1.
- C(N, N) = 1.
- C(N, K) = C(N-1, K-1)+C(N-1, K).

Faça um programa, utilizando uma função recursiva, para calcular C(N,K).

Entradas:

- Argumentos N e K de C.

Saídas

- C(N,K).

Exemplos de Entradas e Saídas:

Entradas:
10 7

Saídas:
120

Entradas:
11 3

Saídas:
165

Peso: 1

Nova Resposta:

Selecione o arquivo com o código fonte do programa que resolve o problema para enviá-lo.

Escolher Arquivo nenhum arquivo selecionado Enviar Resposta

Questão 6: Recursividade - Imprimir inverso sem vetor

Faça um programa que leia uma quantidade de números inteiros e os escreve em ordem inversa, sem usar vetor, por meio de uma função recursiva.

É permitido colocar operações de leitura e escrita na função recursiva.

Entradas:

1. Um número inteiro indicando a quantidade de valores,
2. vários valores inteiros

Saídas:

1. Os valores da entrada, em ordem inversa.

Exemplo de Entradas:

```
15
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1
```

Exemplo de Saída:

```
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
```

Peso: 1

Nova Resposta:

Selecione o arquivo com o código fonte do programa que resolve o problema para enviá-lo.

Escolher Arquivo nenhum arquivo selecionado Enviar Resposta

Questão 7: Recursividade - Funções mutuamente recursivas

Escreva um programa que calcula o valor da função F, definida recursivamente:

F(x) = 1, se x igual a 0,
F(x) = 2H(x) + F(x-1), se x maior que 0 e par,
F(x) = 2H(x) - F(x-1), se x maior que 0 e ímpar.

Observe que F é definida em termos de H, apresentada abaixo:

H(x) = 0, se x igual a 0,
H(x) = H(x-1) + F(x-1), se x maior que 0.

Entradas:

1. Valor inteiro x positivo para o qual se deseja calcular o valor de F(x).

Saídas:

1. Valor de F(x)

```
Exemplo de Entrada:
8

Exemplo de Saída:
2705

======================
```

```
Exemplo de Entrada:
5

Exemplo de saída:
73
```

Peso: 1

Nova Resposta:

Selecione o arquivo com o código fonte do programa que resolve o problema para enviá-lo.

Escolher Arquivo nenhum arquivo selecionado Enviar Resposta

Questão 8: Recursividade - Sem adição

Considere que um determinado sistema numérico fictício não possua a operação de adição. Desenvolva uma função recursiva que calcule a soma de dois números inteiros X e Y. Ao implementar esta função recursiva é permitido utilizar outras duas funções auxiliares não recursivas, chamadas sucessor e antecessor. A função sucessor é responsável por receber um número inteiro N e retornar o valor deste número incrementado em 1, enquanto a função antecessor é responsável por receber um número inteiro N e retornar o valor deste número decrementado em 1. Note que apenas estas duas funções auxiliares (sucessor e antecessor) podem utilizar os operadores de incremento (++) e decremento (--). Seu programa **não** pode utilizar estruturas de repetição (for, while e do-while).

Entradas:

1. Um inteiro X.
2. Um inteiro Y.

Saídas:

1. Um inteiro que indica o valor de X + Y

Exemplo de entrada:

```
2 3
```

Exemplo de saída:

```
5
```

Exemplo de entrada:

```
7 -15
```

Exemplo de saída:

```
-8
```

Peso: 1

Nova Resposta:

Selecione o arquivo com o código fonte do programa que resolve o problema para enviá-lo.

Escolher Arquivo nenhum arquivo selecionado Enviar Resposta

Questão 9: Recursividade - Menor do Vetor

Faça um programa que recebe um vetor V de N elementos e determina, de forma recursiva, o menor elemento do vetor. Podemos usar a seguinte ideia:

- O menor elemento de um vetor de uma única posição é o seu único elemento.
- O menor elemento de um vetor de mais de uma posição é o menor entre o primeiro elemento e o menor do restante do vetor.

Entradas:

- Tamanho N do vetor.
- Elementos do vetor.

Saídas

- Menor elemento do vetor.

Exemplos de Entradas e Saídas:

Entradas:
4
2 4 3 1

Saídas:
1

Peso: 1

Nova Resposta:

Selecione o arquivo com o código fonte do programa que resolve o problema para enviá-lo.

Escolher Arquivo nenhum arquivo selecionado Enviar Resposta

Questão 10: Recursividade - Caminho na Matriz

Dada uma matriz para representar posições no espaço, queremos encontrar um caminho entre duas posições. Encontrar um caminho significa encontrar posições por onde é possível passar.

Este processo pode ser implementado com facilidade usando recursão. Para simplificar ainda mais uma implementação, vamos supor que só é possível caminhar para a direita ou para baixo na matriz. Podemos descrever um caminho de uma origem para um destino assim:

- O caminho é mover-se para posição da direita e depois percorrer o caminho da posição da direita até o destino, se é possível mover-se para direita e se o caminho da posição da direita até o caminho existe.
- O caminho é mover-se para posição de baixo e depois percorrer o caminho da posição de baixo até o destino, se a alternativa anterior não resolveu o problema, se é possível mover-se para baixo e se o caminho da posição de baixo até o destino existe.

A estratégia recursiva acima não tem um caso base, isso fica para você determinar. Note que a estratégia está baseada em tentar à direita antes de tentar abaixo.

Faça um programa que lê uma matriz 10x15 de caracteres representando um lugar. Nessa matriz haverá o caractere '.' (ponto) nas posições em que é possível transitar e haverá o caractere '#'

(cerquilha) nas posições em que não é possível transitar (obstáculos). Encontre na matriz um caminho do canto superior esquerdo até o canto inferior direito usando a estratégia acima. O programa deve alterar a matriz, alterando todas as posições analisadas com um 'x' (xis minúsculo), o que permite não apenas ver o caminho encontrado, mas também quanto processamento foi necessário para encontrá-lo.

Seu programa deverá ter uma função recursiva que encontra caminhos numa matriz e faz as marcações das posições analisadas. Planeje os parâmetros com cuidado. Não é permitido o uso de variáveis globais.

A função principal deve escrever a matriz com as posições marcadas depois do caminho ter sido encontrado.

Exemplo de Entrada:

```
......#........
......###..####
###.........#..
..#.#######.###
..#.#.....#.#..
..#.#.....#.#..
..#.#.....#.#..
###.#########..
............###
..............
```

Exemplo de Saída:

```
xxxxxx#........
...xxx###..####
###xxxxxxxxx#..
..#x#######x###
..#x#.....#x#..
..#x#.....#x#..
..#x#.....#x#..
###x#########..
...xxxxxxxxx###
...........xxxx
```

Peso: 1

Nova Resposta:

Selecione o arquivo com o código fonte do programa que resolve o problema para enviá-lo.

Escolher Arquivo nenhum arquivo selecionado Enviar Resposta

Desenvolvido por Bruno Schneider a partir do programa original (Algod) de Renato R. R. de Oliveira.