Dredd - Juiz Online

Minutos Restantes: 1108

Usuário: Wesley de Jesus Santos

Notas:
Q1: 100
Q2: ?
Q3: ?
Q4: 100
Q5: 100
Q6: 88.5
Q7: ?
Q8: ?
Q9: ?
Q10: ?
Q11: ?
Q12: ?
Total: 32

IAlg EAD - Exercícios de Modularização - 2019/1 - 15/04 à 21/04

Prova Aberta Até: 27/04/2019 06:00:00

Número Máximo de Tentativas: 6

Atenuação da Nota por Tentativa: 0%

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

Contabilização de frequência - Modularizacao

Os exercícios dessa semana que serão utilizados para a contabilização de frequência são: 4 e 11.

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: Modularização - Reverso de um número

Faça um programa que receba um dado valor inteiro e retorne o reverso desse valor. Sendo o número reverso formado pela troca de posição entre a ultima casa com a primeira, e penúltima com a segunda e assim por diante.

O reverso precisa ser criado dentro de uma função, mas sua impressão será executada no programa principal.

Obs: soluções em Python devem possuir um subprograma chamado **principal**.

Entradas:

1. Número inteiro que se deseja obter o reverso.

Saídas:

1. Reverso do valor informado pelo usuário.

Exemplo de Entrada:

```
172
```

Exemplo de Saída:

```
271
```

Exemplo de Entrada:

```
1459
```

Exemplo de Saída:

```
9541
```

Peso: 1

Última tentativa realizada em: 21/04/2019 12:14:49

Tentativas: 1 de 6

Nota (0 a 100): 100

Status ou Justificativa de Nota: Nenhum erro encontrado.

Ver Código da Última Tentativa

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: Modularização – Sequência de Fibonacci

Crie um algoritmo que imprima os **n** primeiros termos da sequência de Fibonacci, utilizando uma função que retorna o n-ésimo termo da referida série. Os termos dessa série são definidos da seguinte forma:

Fib(0) = 0
Fib(1) = 1
Fib(n) = Fib(n-1) + Fib(n-2), para **n** maior ou igual à 2

Obs: para soluções em Python deve existir um subprograma chamado **principal**.

Entradas:

1. Um número inteiro (n).

Saídas:

1. Sequência de Fibonacci começando do 0.

Exemplo de Entradas:

```
10
```

Exemplo de Saída:

```
0 1 1 2 3 5 8 13 21 34
```

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: Modularização - Números Harshad

Na matemática, um número Harshad é um número inteiro que é divisível pela soma dos seus dígitos. O número 81, por exemplo, é um número Harshad pois 8 + 1 = 9, e 81 % 9 = 0.
Escreva uma função que receba um inteiro, e retorne **um valor lógico** que diz se ele é um número Harshad ou não. O processamento dessa função deverá chamar **outra função** que receberá um inteiro e deverá retornar **a soma de seus dígitos**.
Utilize ambas funções em um programa que terá como entrada um número (inteiro e positivo), e deverá exibir como saída o valor lógico retornado pela função.
Dica: transforme o número em uma string para obter a soma dos dígitos.

Obs: soluções em Python devem possuir um subprograma chamado **principal**.

Entradas:

1. Um número inteiro e positivo.

Saídas:

1. Um valor lógico que diz se é um número Harshad ou não.

Exemplo de Entrada:

```
81
```

Exemplo de Saída:

```
True
```

Exemplo de Entrada:

```
143
```

Exemplo de Saída:

```
False
```

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: Modularização - Juros Compostos

Juros compostos são aplicações de juros sobre juros, isto é, os juros compostos são aplicados montante de cada período. Para entender melhor, veja como fica a aplicação mês a mês dos juros:

Primeiro mês: M= C x (1+i)

Segundo mês: M= C x(1+i) x (1+i)

Para simplificar, obtemos a formula a seguir: M = C x (1+i)^t

Em que M é o montante final. i é a taxa de juros aplicada. C é a capital ou valor inicial. t é o tempo de investimento

Entendido isso, faça um programa que receba os valores do usuário de C, i, t. depois faça uma função que pega esses valores e faça as devidas contas, retornando um float como resultado

Entradas:

1. Um C (float) como o valor inicial
2. Um i (float) como a taxa de juros aplicada
3. Um t (inteiro) que indica quantos meses os juros compostos devem ser aplicados

Saídas:

1. O Valor após todas as operações

Exemplo de Entrada:

```
3500.00
2.7
12
```

Exemplo de Saída:

```
4818.52
```

Peso: 1

Última tentativa realizada em: 21/04/2019 12:05:57

Tentativas: 1 de 6

Nota (0 a 100): 100

Status ou Justificativa de Nota: Nenhum erro encontrado.

Ver Código da Última Tentativa

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: Modularização - Número capicua

Um número é dito ser capicua quando lido da esquerda para a direita é o mesmo que quando lido da direita para a esquerda. O ano 2002, por exemplo, é capicua. Implemente uma função que receba como parâmetro **um inteiro** e retorne um valor lógico que indica se o número tem essa característica.

A função será parte de um programa cujo módulo principal lê o número a ser testado e escreve uma resposta baseada no retorno da função.

Entrada do programa:

1. Número inteiro a ser testado.

Saída do programa:

1. Uma palavra ("sim" ou "nao" - em minúsculas, sem acentos) que indica se o número é capicua.

Exemplo de entrada do programa:

```
2002
```

Exemplo de saída do programa:

```
sim
```

Peso: 1

Última tentativa realizada em: 21/04/2019 22:48:46

Tentativas: 1 de 6

Nota (0 a 100): 100

Status ou Justificativa de Nota: Nenhum erro encontrado.

Ver Código da Última Tentativa

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: Modularização - Qual é o mês?

Escreva um programa que tem uma função que receba um número inteiro e retorne o mês correspondente ao número. Por exemplo, **2** corresponde à "**fevereiro**". A função deve retornar a string "**erro**" caso o número recebido não faça sentido. O módulo principal deve chamar a função que resolve o nome do mês e escrever a string retornada.

Os nomes dos meses e a string especial de erro devem usar somente letras minúsculas, sem acentuação nem caracteres

especiais.

Obs: soluções em Python devem possuir um subprograma chamado **principal**.

Entradas:

1. Numero inteiro que supostamente representa um mês do ano.

Saídas:

1. O nome do mês retornado pela função, ou a mensagem de erro.

Exemplo de entrada 1:

```
3
```

Exemplo de saída 1:

```
marco
```

Exemplo de entrada 2:

```
15
```

Exemplo de saída 2:

```
erro
```

Peso: 1

Última tentativa realizada em: 21/04/2019 23:07:50

Tentativas: 1 de 6

Nota (0 a 100): 88.5

Status ou Justificativa de Nota: Existe um trecho perigoso no código. A quantidade de dados escritos pelo programa é diferente da quantidade de dados esperados.

Ver Código da Última Tentativa

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: Modularização - Divisores de um número

Escreva um programa que tem uma função que recebe um número inteiro e mostre na tela os seus divisores, e também retorne: a) o seu segundo menor divisor; e b) o seu segundo maior divisor.

Por exemplo, para o número 21 a função deve verificar e mostrar os números 1, 3, 7 e 21, que são todos os seus divisores, e também retornar para o programa principal os números 3 e 7, pois o número 3 é o segundo menor divisor de 21, e o número 7 é o segundo maior divisor de 21.

O programa principal deve utilizar a função e mostrar, na tela do monitor, os números do segundo menor divisor e do segundo maior divisor, em uma única linha.

Observações:

1. A impressão dos divisores deve ser feita dentro da função;
2. A impressão do segundo menor divisor e do segundo maior divisor deve ser feita no programa principal;
3. Atente para o caso dos números primos, em que o menor divisor é 1 e o maior divisor é o próprio número.
4. Para soluções em Python deve existir um subprograma chamado **principal**.

Entradas:

1. Um número inteiro

Saídas:

1. Todos os divisores
2. O segundo menor divisor
3. O segundo maior divisor

Exemplo de entrada:

```
21
```

Exemplo de saída:

```
1
3
7
21
3 7
```

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: Modularização - Somatório, combinação e fatorial

Faça um programa que receba no usuário um número inteiro positivo e não nulo do usuário (n ≥ 1) e calcule e exiba o valor da seguinte expressão:

$$\sum_{k=1}^{n} C_{n,k} \quad , \text{ sendo que } \quad C_{n,k} = \frac{n!}{k!(n-k)!}$$

Com essas expressões em mente, seu programa deverá ser feito utilizando modularização, sendo criada uma função "somatório", que calcula o somatório das combinações da expressão, uma função "combinação", que calcula cada combinação pedida, e uma função "fatorial" para o cálculo de cada fatorial dentro da combinação.

Organize seu programa de forma que cada função dependa uma da outra, ou seja, o cálculo da expressão seja composto por várias chamadas de funções (uma função chamando a outra).

Entrada:

- Um número inteiro positivo não nulo (n ≥ 1).

Saída

- Um número inteiro - o valor do cálculo da expressão (somatório).

Exempo de Entrada:

```
5
```

Exemplo de Saída

```
31
```

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: Modularização - Números primos em um intervalo

Faça um programa que recebe dois números inteiros N e M, N ≤ M, e calcula quantos números primos existem nesse intervalo. O cálculo da quantidade de primos deve ser feito em uma função, que tem como parâmetros os valores de início e fim do intervalo, N e M, respectivamente. A leitura e a saída dos dados devem ser realizadas no subprograma principal.

Entradas:

1. Dois valores inteiros N e M, N ≤ M (na mesma linha).

Saídas:

1. A quantidade de números primos no intervalo [N, M].

Exemplo de entrada:

```
3 13
```

Exemplo de saída:

```
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 10: Modularização – Somatório com alternância de sinal

$$1^1 - 2^2 + 3^3 - 4^1 + 5^2 - 6^3 + 7^1 - 8^2 + 9^3 - ...$$

Escreva um programa que calcula uma certa quantidade de termos da série acima. Esta quantidade é uma entrada do programa e deve ser um número inteiro maior que zero. Se o usuário informar um número inteiro menor que 1, o programa deve ler um novo valor, repetidamente, até que o valor seja maior que zero.

O programa deve ter uma função que recebe o número de termos e retorna o somatório dos termos.

As operações de leitura e escrita devem estar na função principal. Para programas em Python, deve ser criada uma função cujo nome é "principal".

Entradas:

1. A quantidade de termos a ser somada (número inteiro maior que zero, sujeito a releitura, conforme explicado acima)

Saídas:

1. O valor do somatório

Exemplo de Entrada 1:

```
4
```

Exemplo de Saída 1:

```
20
```

Exemplo de Entrada 2:

```
-5
25
```

Exemplo de Saída 2:

```
-7895
```

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 11: Modularizacao - Palindromos

Um palíndromo é um texto que pode ser lido tanto da esquerda para a direita como da direita para a esquerda. Por exemplo, sem considerar sinais gráficos, acentos e letras maiúsculas e minúsculas: "Socorram-me, subi no ônibus em Marrocos".

Implemente uma função que receba como parâmetro **uma palavra** e retorne um valor lógico que indica se a palavra tem essa característica. Uma outra função deve receber todas as palavras que são palíndromos e devolver ao módulo principal a maior e a menor delas.

As funções serão parte de um programa cujo módulo principal lê a quantidade de palavras a serem analisadas e escreve aquelas que não são palíndromos, e dentre os palíndromos, a menor e a maior

delas (a menor é a primeira das palavras quando ordenadas em ordem alfabética, e a maior é a última). Suponha que sempre existirá ao menos uma palavra palíndromo, possibilitando a determinação do maior e do menor.

Entradas:

1. A quantidade de palavras a serem testadas.
2. Palavras a serem testadas (uma em cada linha).

Saídas:

1. Palavras que não são palíndromos;
2. menor palavra palíndromo lida (suponha que sempre existe uma);
3. maior palavra palíndromo lida (suponha que sempre existe uma).

Exemplo de entrada:

```
7
casa
ovo
pelo
tomate
arara
papa
ama
```

Exemplo de saída:

```
casa
pelo
tomate
papa
ama
ovo
```

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 12: Modularização – 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 $\geq$ 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