Lógica de Programação

Aula 12 - Estruturas de Repetição – Exercícios

Apresentação da Aula

Na aula anterior, você conheceu as estruturas homogêneas unidimensionais (*Arrays*) e, nesta aula, você exercitará o uso das estruturas de dados homogêneas, tendo a oportunidade de construir algoritmos que utilizam esse elemento (vetor) para solucionar problemas.

Objetivos

- Exercitar a utilização das estruturas de dados homogêneas unidimensionais.

Atividade 01

Preenchendo e Lendo um Vetor I

Construa um algoritmo que receberá 10 valores inteiros e escreva, um abaixo do outro, os valores na ordem **inversa** em que foram lidos, ou seja, o último valor lido será o primeiro valor impresso.

Atividade 02

Preenchendo e Lendo um Vetor II

Construa um algoritmo que receberá 30 valores inteiros.

Ao final, seu algoritmo devá apresentar os números pares em uma linha e na linha seguinte os números ímpares, na ordem em que foram lidos.

Se os valores recebidos pelo algoritmo são:

```
-30, -55, 28, -23, -93, 80, 90, 75, -7, -2, 96, 57, -24, -46, -66,
87, 74, -39, 21, -81, -6, -65, 27, 94, -42, -1, -50, -94, -71 e -91.
```

Ele deverá apresentar o resultado no seguinte formato:

```
-30, 28, 80, 90, -2, 96, -24, -46, -66, 74, -6, 94, 42, -50, -94,
-55, -23, -93, 75, -7, 57, 87, -39, 21, -81, -65, 27, -1, -71, -91.
```

Obs.: Há um espaço em branco a direita de cada vírgula.

Atividade 03

Menor e Posição

Construa um algoritmo que receberá 100 valores inteiros. Ao final, o algoritmo irá imprimir o menor valor recebido. O índice em que o menor valor está armazenado. Por último, os valores armazenados nas posições imediatamente anteriores e posteriores ao menor elemento encontrado. Obedeça o seguinte formato para apresentar as informações:

```
Menor valor recebido: 10
Índice do menor valor: 19
Valor armazenado na posição anterior [18]: 12
Valor armazenado na posição posterior [20]: 34
```

Caso o menor valor esteja na primeira posição, seu algoritmo não deve imprimir a linha correspondente ao elemente anterior e se estiver na última posição, não deve imprimir o elemento posterior.

Atividade 04

Preenchimento de Vetor I

Construa um algoritmo com um vetor de 100 posições. Esse algoritmo irá receber uma quantidade indefinida de valores inteiros e armazena-los no vetor. Quando o valor 0 (zero) for recebido pelo seu algoritmo, interrompa o recebimento de novos valores (e não armazene o 0 (zero)).

Em seguida, o seu algoritmo deverá verificar se os elementos armazenados nos índices ímpares são divisíveis pelos elementos imediatamentes posteriores (índice+1).

Por exemplo, se o os valores 38, 16, 29, 34, 33, 3, 35, 7, 19, 30 e 0 são recebidos pelo seu algoritmo, ele terá como saída o seguinte resultado:

```
Conteúdo do vetor:
38, 16, 29, 34, 33, 3, 35, 7, 19, 30,
38 não é divisível por 16
29 não é divisível por 34
33 é divisível por 3
35 é divisível por 7
19 não é divisível por 30
```

Obs.: Há um espaço em branco a direita de cada vírgula.

Atividade 05

Preenchendo e Lendo um Vetor III

Construa um programa com 2 vetores, cada um de 10 posições. O seu algoritmo receberá 15 números inteiros. Os números ímpares deverão ser armazenados no primeiro vetor. Já os pares serão armazenados no segundo vetor. Se um vetor ficar cheio (completar as 10 posições) durante a leitura dos valores, você deverá "descartar" o valor mais antigo (a primeira posição do vetor) e mover as posições subsequentes em "uma casa para esquerda" para que seja possível armazenar o 10º valor. Repita essa operação quantas vezes for necessário, até que todos os 15 valores sejam lidos pelo seu programa.

Ao final, o algoritmo deverá imprimir os números armazenados no primeiro vetor em uma linha, e os armazenados no segundo vetor na linha imediatamente abaixo. Os números devem estar separados por um espaço em branco simples.

Atividade 06

Seleção em Vetor I

Construa um algoritmo que irá receber e armazenar 20 números reais.

Em seguida, o seu programa deverá escrever a média aritmética de todos os valores. Na linha seguinte, imprima todos que estão acima da média. Na terceira linha, imprima os números que estão abaixo da média. Os números devem estar separados por um espaço em branco simples (a sua direira).

Para a entrada 51.2, 39.83, 86.71, 78.72, 71.80, 82.0, 90.87, 43.9, 60.76, 4.12, 80.10, 49.29, 0.42, 20.95, 72.88, 16.52, 65.3, 92.78, 27.74 e 56.4, será apresentada a saída:

```
54.6145
86.71 78.72 71.8 82.0 90.87 60.76 80.1 72.88 65.3 92.78 56.4
51.2 39.83 43.9 4.12 49.29 0.42 20.95 16.52 27.74
```

Obs.: Números iguais a média não serão impressos.

Atividade 07

Seleção em Vetor II

Construa um algoritmo que recebe e armazena até 100 números inteiros. Quando o valor 0 (zero) for recebido pelo seu algoritmo, interrompa o recebimento de novos valores (e não armazene o 0 (zero)).

Em seguida, imprima os números lidos em uma linha, separados por espaços em branco. O seu programa deverá procurar os valores que são menores do que zero e inverter o sinal para que fiquem positivos. Além disso, os números deverão ser impressos na ordem inversa que foram lidos. Por exemplo, se o vetor lido for:

```
11 5 -4 10 -2 1 9 16 57 -19
```

O segundo vetor impresso será:

```
19 57 16 9 1 2 10 4 5 11
```

Obs.: Há um espaço em branco a direita de cada número.

Atividade 08

Divisão no Vetor

Construa um algoritmo que recebe e armazena até 100 números inteiros. Quando o valor 0 (zero) for recebido pelo seu algoritmo, interrompa o recebimento de novos valores (e não armazene o 0 (zero)).

Em seguida, o seu programa deverá imprimir todos os números que são divisíveis pelo último número armazenado no vetor. Escreva os valores na ordem inversa em que foram lidos. Por exemplo, para os números armazenados no vetor:

```
11 5 4 10 2 1 9 16 57 19 3
```

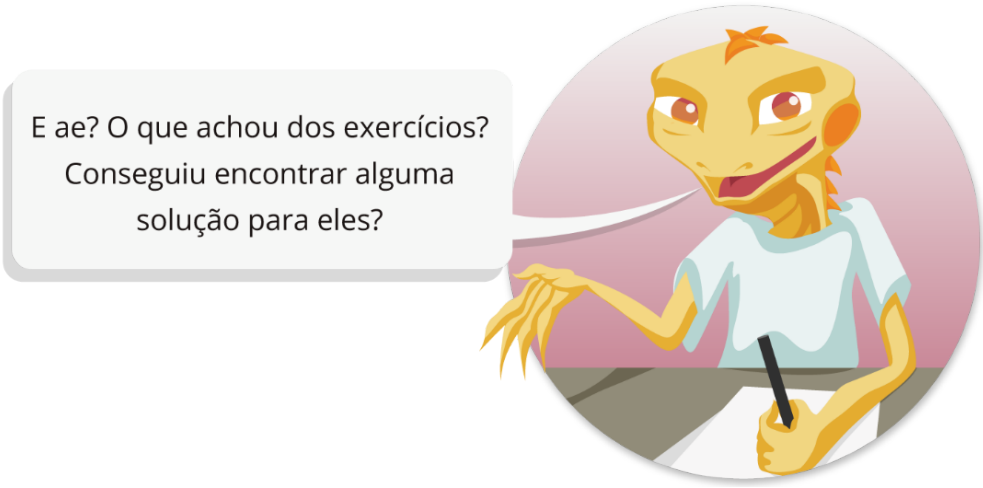
Serão impressos, na ordem inversa, os números divisíveis por 3 (último número armazenado):

```
57
9
```

Atividade 09

Ordenação de Um Vetor

Construa um algoritmo que receberá 100 números inteiros. Utilizando apenas um único vetor, o seu algoritmo deverá ordenar os valores lidos em ordem crescente. Ao final, o programa deverá realizar a impressão dos números ordenados (um por linha).



Resumo

Nesta aula, você exercitou o uso das estruturas de dados homogêneas e trabalhou com as estruturas unidimensionais, construindo algoritmos por meio delas e das estruturas de repetição. Nas próximas aulas, você conhecerá as estruturas homogêneas bidimensionais. Portanto, caso tenha encontrado alguma dificuldade, converse com seu mediador para poder sanar o quanto antes as suas dúvidas. O completo entendimento das estruturas homogêneas unidimensionais é importante para a construção dos seus próximos conhecimentos sobre a construção de algoritmos.

Referências

Linguagem Potigol: Programação para todos. Disponível em: <http://potigol.github.io/>. Acesso: 14 jun. 2018.

1173 - Preenchimento de Vetor I – URI Online Judge. Disponível em: <https://www.urionlinejudge.com.br/judge/pt/problems/view/1173>. Acesso: 14 jun. 2018.

1174 – Seleção em Vetor I – URI Online Judge. Disponível em: <https://www.urionlinejudge.com.br/judge/pt/problems/view/1174>. Acesso: 16 jun. 2018.

1176 – Fibonacci em Vetor – URI Online Judge. Disponível em: <https://www.urionlinejudge.com.br/judge/pt/problems/view/1176>. Acesso: 18 jun. 2018.

1179 - Preenchimento de Vetor IV – URI Online Judge. Disponível em: <https://www.urionlinejudge.com.br/judge/pt/problems/view/1179>. Acesso: 15 jun. 2018.

1180 - Menor e Posição – URI Online Judge. Disponível em: <https://www.urionlinejudge.com.br/judge/pt/problems/view/1180>. Acesso: 14 jun. 2018.