FEELT31201 - Programação Procedimental - 2022/1

Laboratório 03

Prazo: 04/11 - 23:59h

Básico 1

1.25 pontos

- Nome do arquivo com código-fonte: **`minMax.c`**
- Tarefa a cumprir:

  Peça ao usuário a quantidade de elementos do tipo inteiro (N) que deseja trabalhar. Depois peça elemento a elemento (a_i) quais os números desejados. Imprima os números informados, mas, logo após o menor deles, imprima o caractere `<` e, logo após o maior deles, imprima o caractere `>`.
- Exemplo: se o usuário informar que deseja trabalhar com 6 números que são 3, 1, 2, 42, 25 e 12, imprima `3 1< 2 42> 25 12`.
- Método sugerido para entrada de dados pelo usuário:

```
// mensagem para o usuário
scanf("%d", &N);
for(int i=0; i < N; i++) {
    // mensagem para o usuário
    scanf("%d", a+i);
}
```

- Casos de teste:

Entrada		Saída esperada contém
6	3 1 2 42 25 12	`3 1< 2 42> 25 12`
4	24 12 -12 -24	`24> 12 -12 -24<`
3	1 2 3	`1< 2 3>`
9	2 1 6 8 4 7 5 3 8	`2 1 6 8> 4 7 5 3 0< 8>`

- *Dica(s):* Você deve utilizar um vetor para armazenar as entradas do usuário individualmente. Você pode usar o especificador `%g` para deixar o programa escolher o número de casas decimais para mostrar ao usuário.

Básico 2

1.25 pontos

- Nome do arquivo para o código-fonte: **`quaisDiv.c`**
- Tarefa a cumprir:

  Peça ao usuário a quantidade de elementos do tipo inteiro (N) que deseja trabalhar. Depois peça elemento a elemento (a_i) quais os números desejados. Na sequência, peça o inteiro que será a referência (R) para encontrar no vetor quem é seu divisor. Imprima os números informados, mas, logo após um divisor de R, imprima o caractere `#`.

- Exemplo: se o usuário informar que deseja trabalhar com 6 números que são 3, 1, 2, 42, 25 e 12, e quer o 12 como referência, imprima `3# 1# 2# 42 25 12#`.
- Método de entrada do usuário:

```
// mensagem para o usuário
scanf("%d", &N);
for(int i=0; i < N; i++) {
    // mensagem para o usuário
    scanf("%d", a+i);
}
// mensagem para o usuário
scanf("%d", &R);
```

- Casos de teste:

Entrada do usuário	A saída esperada contém
6 3 1 2 42 25 12 12	`3# 1# 2# 42 25 12#`
7 1 2 4 8 16 32 64 64	`1# 2# 4# 8# 16# 32# 64#`
3 3 13 123 121	`3 13 123`
4 3 331 2 221 662	`3 331# 2# 221`

- **Dicas:** Você deve utilizar um vetor para armazenar as entradas do usuário individualmente. Você sabe se um número n é divisor (ou divide) um número m se e somente se `m % n == 0`.

Básico 3

1.25 pontos

- Nome do arquivo para o código-fonte: **`somatorio.c`**
- Tarefa a cumprir:

  Peça ao usuário o número de elementos do tipo ponto flutante (N) que deseja trabalhar. Depois peça elemento a elemento (a_i) quais os números desejados. Imprima os elementos informados separados pelo sinal de soma ($+$) seguidos pelo sinal de igual ($=$) e terminando com o somatório dos elementos.

- Exemplo: se o usuário informar que quer trabalhar com 6 elementos, informando na sequência: 1, 2, -3, 4, 6.25 e 12. Como resposta, teremos `1 + 2 + -3 + 6 + 6.25 + 12 = 24.25`.
- Método de entrada do usuário:

```
// mensagem para o usuário
scanf("%d", &N);
for(int i=0; i < N; i++) {
    // mensagem para o usuário
    scanf("%f", a+i);
}
```

- Casos de teste:

Entrada do usuário	A saída esperada contém
6 1 2 -3 6 6.25 12	`1 + 2 + -3 + 6 + 6.25 + 12 = 24.25`
2 5 -5	`5 + -5 = 0`
4 17 14 1 2	`17 + 14 + 1 + 2 = 34`
4 1.5 3.2 8.9 11.1	`1.5 + 3.2 + 8.9 + 11.1 = 24.7`

- *Dicas:* Você deve utilizar um vetor para armazenar as entradas do usuário individualmente. Use o especificador `%g` para mostrar os números.

Básico 4

1.25 pontos

- Nome do arquivo para o código-fonte: **`produtorio.c`**
- Tarefa a cumprir:

  Peça ao usuário a quantidade de elementos do tipo ponto flutante (N) que deseja trabalhar. Depois peça elemento a elemento (a_i) quais os números desejados. Imprima os elementos informados separados pelo sinal de multiplicação ($*$) seguidos pelo sinal de igual ($=$) e terminando com o produtório dos elementos.
- Exemplo: se o usuário informar que quer trabalhar com 6 elementos, informando na sequência: 1, 2, -3, 4, 6.25 e 12. Como resposta, teremos `1 * 2 * -3 * 6 * 6.25 * 12 = -2700`.
- Método de entrada do usuário:

```
// mensagem para o usuário
scanf("%d", &N);
for(int i=0; i < N; i++) {
    // mensagem para o usuário
    scanf("%f", a+i);
}
```

- Casos de teste:

Entrada do usuário	A saída esperada contém
6 1 2 -3 6 6.25 12	`1 * 2 * -3 * 6 * 6.25 * 12 = -2700`
2 5 -5	`5 * -5 = -25`
4 17 14 1 2	`17 * 14 * 1 * 2 = 476`
4 1.5 3.2 8.9 11.1	`1.5 * 3.2 * 8.9 * 11.1 = 474.192`

- *Dicas:* Você deve utilizar um vetor para armazenar as entradas do usuário individualmente. Use o especificador `%g` para mostrar os números.

Médio 1

1.5 pontos

- Nome do arquivo para código-fonte: `muSigma.c`
- Tarefa a cumprir:

  Peça ao usuário o número de elementos do tipo ponto flutante (N) que deseja trabalhar. Depois peça elemento a elemento (a_i) quais os números desejados. Calcule e apresente a média (μ) e o desvio-padrão (σ, assuma a versão com base na população) da coleção de números:

$$\mu = \frac{1}{N} \cdot \sum_{i=0}^{N-1} a_i$$

$$\sigma = \sqrt{\frac{1}{N} \cdot \sum_{i=0}^{N-1} (a_i - \mu)^2}$$

- Exemplo: se o usuário informar que deseja 5 números e entrar com os números 2.3, 5.25, -0.5, 3.0, 4.7, então a média μ será $(2.3+5.25+(-0.5)+3.0+4.7)/5 = 14.75/5 = 2.95$ e o desvio-padrão σ será $((-0.65^2 + 2.3^2 + (-3.45)^2 + 0.05^2 + 1.75^2)/5)^{0.5} \approx 2.034$.
- Método de entrada do usuário:

```
// mensagem para o usuário
scanf("%d", &N);
for(int i=0; i < N; i++) {
    // mensagem para o usuário
    scanf("%d", a+i);
}
```

- Casos de teste:

Entrada do usuário	A saída esperada contém
5 2.3 5.25 -0.5 3 4.7	média `2.95`, desvio `2.03...`
6 9 9 9 1 1 1	média `5`, desvio `4`
2 42 43	média `42.5`, desvio `0.5`
10 1 2 3 4 5 6 7 8 9 9.75	média `5.475`, desvio `2.83...`

- *Dicas:* Você deve pensar em duas funções separadas, uma para média, outra para desvio-padrão, assumindo como argumento ambos o endereço do início do vetor e a quantidade de elementos nele.

Médio 2

1.5 pontos

- Nome do arquivo para o código-fonte: **`remNegativos.c`**
- Tarefa a cumprir:

  O usuário deve entrar com a quantidade de elementos desejados para um vetor e depois alimentar cada um deles entre inteiros positivos e negativos. Escreva um programa que remova todos os números negativos, deixando apenas os positivos e zeros. Imprima o vetor se contiver algum elemento. Se todos os números forem negativos, avise ao usuário que o vetor está vazio.

- Exemplo: o usuário informa 4 elementos e então alimenta '-1 2 -3 4' no vetor. O programa remove todos os números negativos do vetor, deixando: `2, 4`.
- Método de entrada do usuário:

```
// mensagem para o usuário
scanf("%d", &nelem);
// mensagem para o usuário
for(int i = 0; i < nelem; i++)
    scanf("%d", vetor + i);
```

- Casos de teste:

Entrada do usuário	A saída esperada contém
4 -1 2 -3 4	`2, 4`
4 -1 -2 -3 -4	`vazio`
4 1 2 3 4	`1, 2, 3, 4`
7 -1 2 -3 4 -5 6 -7	`2, 4, 6`

- *Dicas:* Você deve pensar em uma função para contar os números negativos e retornar o tamanho do vetor resposta, outra para percorrer o vetor e colocar no vetor resposta apenas os números positivos e zeros.

Difícil

2.0 points

- Nome do arquivo para o código-fonte: `diferencas.c`
- Tarefa a cumprir:

  O usuário deve entrar com a quantidade de elementos desejados (N) para um vetor

  $$A = \{a_i\}|_{i=0...N-1}$$

  de números de ponto flutuante e então alimentar cada um deles. Escreva uma função que preencha outro vetor

  $$B = \{b_j\}|_{j=0...N-2}$$

  com a diferença entre elementos adjacentes:

  $$b_j = a_{i+1} - a_i$$

  Além disso, o programa deve apresentar tanto o **mínimo** quanto o **máximo** das diferenças calculadas (vetor B).
- Exemplo: o usuário informa 4 elementos e então alimenta '1.2 2.0 -3.8 4.5' no vetor. O programa calcula as diferenças entre os elementos adjacentes e preenche outro vetor com `-0.8, 5.8, -8.3`. A saída esperada também apresenta as diferenças máximas (`5.8`) e mínimas (`-8.3`).
- Método de entrada do usuário:

```
// mensagem para o usuário
scanf("%d", &N);
// mensagem para o usuário
for(int i = 0; i < N; i++)
    scanf("%f", vetor + i); // para float
    // ou ****
    scanf("%lf", vetor + i); // para double
```

- Casos de teste:

Entrada do usuário	**A saída esperada contém**	min	max
4 1.2 2.0 -3.8 4.5	`0.8, -5.8, 8.3`	`-5.8`	`8.3`
4 3 3 3 3	`0, 0, 0`	`0`	`0`
6 -1.5 -2.0 -2.5 -3.0 -3.5 -4.0	`-0.5, -0.5, -0.5, -0.5, -0.5`	`-0.5`	`-0.5`
8 1 1 2 3 5 8 13 21	`0, 1, 1, 2, 3, 5, 8`	`0`	`8`

- **Dicas:** `https://www.mathworks.com/help/matlab/ref/diff.html`
  Não se esqueça de antes descobrir qual o tamanho do seu vetor de resposta.