Ordenação e Struct

Faça um programa que receba os dados de uma companhia e imprima ordenado de forma decrescente em relação ao valor da ação da companhia.

Dados da companhia:

- nome da companhia (string que aceita espaços)

- valor atual da ação (em reais)

- variação da ação em porcentagem (valor real), ou seja, quanto a ação cresceu ou caiu desde a abertura da bolsa no dia.

O seu programa deve ler os dados de 10 companhias que trabalham com ações. Além disso, deve ter um subprograma que imprime os dados das companhias ordenados pelo valor atual da ação, formando um ranking (decrescente). A ordenação dos dados deve ser feita usando um método selection sort.

Entradas:

- Nome da companhia (string)
- Valor das ações (float)
- Porcentagem em decimais (float)

Saídas:

- O ranking decrescente das companhias

Exemplos de Entradas e Saídas:

Entradas:

Gerdau PN

22.97 -1.16

Gol PN

25.37 2.22

Hapvida ON

17.76 1.43

Hypera ON

33.25 0.57

Iguatemi ON

34.98 0.75

Intermedica ON

97.89 1.37

IRB Brasil On

7.24 1.23

Itau Unibanco PN

28.46 2.13

Itausa PN

10.81 0.19

JBS ON

24.52 1.20

Saídas:

RANKING:

Posicao 1

Companhia: Intermedica ON

Valor: R$ 97.89

Porcentagem: 1.37

Posicao 2

Companhia: Iguatemi ON

Valor: R$ 34.98

Porcentagem: 0.75

Posicao 3

Companhia: Hypera ON

Valor: R$ 33.25

Porcentagem: 0.57

Posicao 4

Companhia: Itau Unibanco PN

Valor: R$ 28.46

Porcentagem: 2.13

Posicao 5

Companhia: Gol PN

Valor: R$ 25.37

Porcentagem: 2.22

Posicao 6

Companhia: JBS ON

Valor: R$ 24.52

Porcentagem: 1.2

Posicao 7

Companhia: Gerdau PN

Valor: R$ 22.97

Porcentagem: -1.16

Posicao 8

Companhia: Hapvida ON

Valor: R$ 17.76

Porcentagem: 1.43

Posicao 9

Companhia: Itausa PN

Valor: R$ 10.81

Porcentagem: 0.19

Posicao 10

Companhia: IRB Brasil On

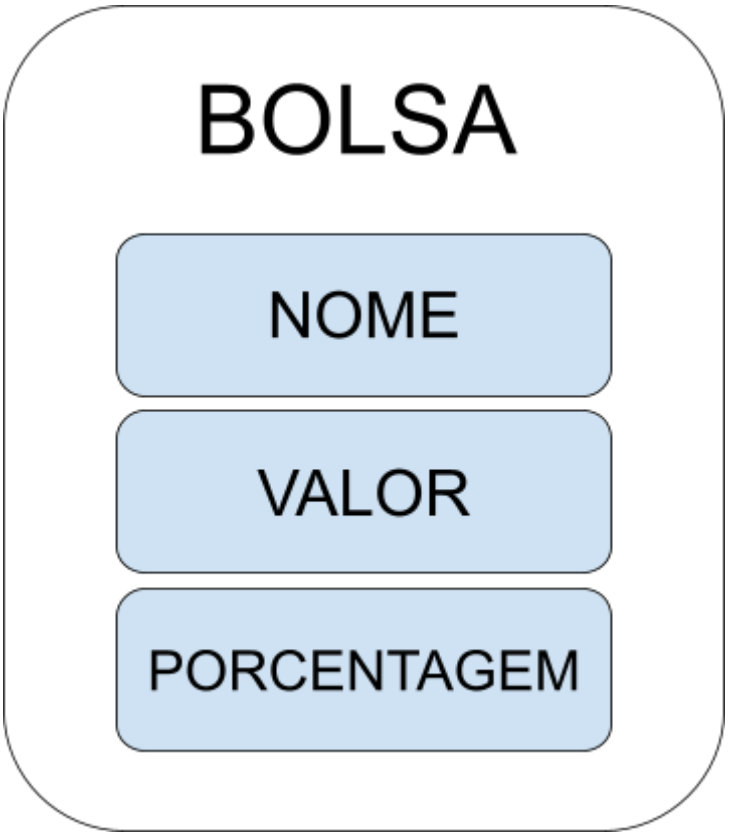
Valor: R$ 7.24

Porcentagem: 1.23

2- SUGESTÃO DE SOLUÇÃO:

Ideia geral:

Para resolver esse problema devemos pensar na variável como um objeto da vida real e esse objeto tem três características: nome, valor e porcentagem. Para representar isso em código, no C++ nós usamos o tipo `struct` para criar um objeto. Então nós criaremos esse objeto com os atributos necessários para suprir as características dele. Logo após isso, nós usaremos um algoritmo de ordenação, para podermos fazer o ranking das companhias e imprimir nosso vetor do objeto ordenado.



Passo 1: Armazenamento dos dados: Para armazenar os dados das 10 companhias, nós criaremos um vetor de 10 posições do nosso objeto, como foi dito anteriormente usaremos o tipo `struct`, e dentro dele iremos armazenar o nome da companhia como `string`, o valor da ação como `float` e a porcentagem como `float`. Como no código a seguir:

```
const int QUANTIDADE_DE_COMPANHIAS = 10;
struct bolsa {
    string companhia;
    float valor;
    float porcentagem;
};
int main() {
    bolsa minhaBolsa[QUANTIDADE_DE_COMPANHIAS];
```

Passo 2: Leitura dos dados de entrada: Para efetuar a leitura dos nossos dados de entrada, criaremos um subprograma que recebe por parâmetro o vetor do objeto, e usa o laço de repetição for, em que a cada interação lê o nome(`string`), o valor da ação(`float`) e a porcentagem(`float`) da companhia naquela posição. Para acessar um atributo de uma `struct` usamos "`objeto.atributo`". Porém o nome da companhia pode ter espaços, para isso usaremos o método `getline()` para ler o nome. O método `getline()` requer atenção, pois se nós lermos qualquer outro valor antes dele e não usarmos o `cin.ignore()`, ele acaba "pegando o lixo" daquela leitura, não agindo da forma que

gostaríamos. Por isso, no final de toda iteração, usaremos o `cin.ignore()`, para que na próxima iteração o `getline()` leia somente o que desejamos, que é o nome da companhia. Como no código a seguir:

```
void leitura( bolsa minhaBolsa[ ] ) {
   for ( int i = 0; i < QUANTIDADE_DE_COMPANHIAS; i++ ) {
      getline( cin, minhaBolsa[i].companhia );
      cin >> minhaBolsa[i].valor >> minhaBolsa[i].porcentagem;
      cin.ignore();
   }
}
```

Passo 3: Ranking(Ordenação): Nosso terceiro passo é fazer o ranking da companhia com maior valor para a companhia com menor valor e imprimir na tela. Para isso usaremos o algoritmo de ordenação Selection Sort. Esse algoritmo, funciona percorrendo do primeiro elemento até o último elemento do vetor. Dependendo da regra de ordenação (crescente ou decrescente), a cada interação ele define o valor de uma variável auxiliar (se for crescente, será posição menor, se for decrescente será posição maior) o valor da posição do vetor naquela interação. Logo após isso ele irá percorrer daquela posição do vetor até o fim do vetor comparando (se for crescente, se o vetor naquela posição tem valor menor que o vetor na posição da variável auxiliar, ou decrescente se o vetor naquela posição tem valor maior que o vetor na posição da variável auxiliar). Caso a condição seja favorável, ele atribui aquela posição ao valor da variável auxiliar. Logo após isso, faz as trocas de valores da posição selecionada e da posição auxiliar. Neste caso, como usaremos um objeto, temos que lembrar que a comparação será somente do atributo valor. Veja o exemplo no código a seguir:

```
void selectionSort( bolsa minhaBolsa[ ] ) {
   bolsa minhaBolsaAUX;
   int posicao_maior;

   for ( int i = 0; i < QUANTIDADE_DE_COMPANHIAS; i++ ) {
      posicao_maior = i;
      for ( int j = i; j < QUANTIDADE_DE_COMPANHIAS; j++ ) {
         if ( minhaBolsa[j].valor > minhaBolsa[posicao_maior].valor ) {
            posicao_maior = j;
         }
      }
      minhaBolsaAUX = minhaBolsa[i];
      minhaBolsa[i] = minhaBolsa[posicao_maior];
      minhaBolsa[posicao_maior] = minhaBolsaAUX;
   }
}
```

Passo 4: Ranking(Saída dos dados): Neste passo, nós faremos a saída dos dados. Para isso faremos um subprograma, neste subprograma chamaremos o algoritmo de ordenação e logo após, faremos a sáida dos dados. Para formatar a saída como solicitado, usaremos "`\t`" para fazer a tabulação, assim deixando os dados alinhados. Veja no código a seguir:

```
void ranking( bolsa minhaBolsa[ ] ) {
   selectionSort( minhaBolsa );
   cout << "RANKING: " << endl;
   for ( int i = 0; i < QUANTIDADE_DE_COMPANHIAS; i++ ) {
      cout << "Posicao " << i + 1 << endl
         << "\tCompanhia: " << minhaBolsa[i].companhia << endl
         << "\tValor: R$ " << minhaBolsa[i].valor << endl
         << "\tPorcentagem: " << minhaBolsa[i].porcentagem << endl << endl;
   }
}
```

Exemplo de código:

```
#include <iostream>
using namespace std;

const int QUANTIDADE_DE_COMPANHIAS = 10;

struct bolsa {
   string companhia;
   float valor;
   float porcentagem;
};


void leitura( bolsa minhaBolsa[ ] ) {
   for ( int i = 0; i < QUANTIDADE_DE_COMPANHIAS; i++ ) {
      getline( cin, minhaBolsa[i].companhia );
      cin >> minhaBolsa[i].valor >> minhaBolsa[i].porcentagem;
      cin.ignore();
   }
}


void selectionSort( bolsa minhaBolsa[ ] ) {
   bolsa minhaBolsaAUX;
   int posicao_maior;

   for ( int i = 0; i < QUANTIDADE_DE_COMPANHIAS; i++ ) {
      posicao_maior = i;
      for ( int j = i; j < QUANTIDADE_DE_COMPANHIAS; j++ ) {
         if ( minhaBolsa[j].valor > minhaBolsa[posicao_maior].valor ) {
            posicao_maior = j;
         }
```

```
        }
        minhaBolsaAUX = minhaBolsa[i];
        minhaBolsa[i] = minhaBolsa[posicao_maior];
        minhaBolsa[posicao_maior] = minhaBolsaAUX;
    }
}

void ranking( bolsa minhaBolsa[ ] ) {
    selectionSort( minhaBolsa );
    cout << "RANKING: " << endl;
    for ( int i = 0; i < QUANTIDADE_DE_COMPANHIAS; i++ ) {
        cout << "Posicao " << i + 1 << endl
             << "\tCompanhia: " << minhaBolsa[i].companhia << endl
             << "\tValor: R$ " << minhaBolsa[i].valor << endl
             << "\tPorcentagem: " << minhaBolsa[i].porcentagem << endl << endl;
    }
}

int main() {
    bolsa minhaBolsa[QUANTIDADE_DE_COMPANHIAS];
    leitura( minhaBolsa );
    ranking( minhaBolsa );
    return 0;
}
```

Vídeo-demonstração: