INSTRUÇÃO PRÁTICA		PI-P017
MÓDULO	PI - PROGRAMAÇÃO IMPERATIVA	
OBJETIVO DA ATIVIDADE		**TEMPO**
Capacitar os alunos a aplicarem de forma prática e eficaz os conceitos fundamentais de programação orientada a objetos em C++, consolidando sua compreensão sobre criação de classes, instanciação de objetos e encapsulamento, além de promover habilidades de resolução de problemas em grupo e o desenvolvimento de soluções complexas utilizando o paradigma orientado a objetos.		2h

DESCRIÇÃO

Exercício 1: Crie um repositório para esta atividade.

- Sua equipe deve criar um repositório no GitHub para esta atividade. O repositório deve conter:
  - Um README descrevendo a atividade e as informações dos membros da equipe
  - Os arquivos de código fonte de cada uma das atividades.
- Para realizar a atividade cada um dos membros da equipe deve criar um branch no repositório, onde fará as implementações da parte individual e outro Branch para mesclar os códigos relativos à parte em grupo.
- Publicar no Moodle o endereço do repositório e o branch específico em que está desenvolvendo suas atividades
- Utilize o fórum do Moodle para tirar suas dúvidas sobre os exercícios.

Parte Individual:

Exercício 2: Criando uma Classe Básica

Crie uma classe chamada Ponto que represente um ponto no plano cartesiano com coordenadas x e y. Inclua construtores, métodos para definir e obter as coordenadas, e um método para calcular a distância até a origem (0,0).

Para testar sua classe, se baseie nos exemplos abaixo:

```
Ponto p1(3, 4);
double distancia = p1.calcularDistancia();
// Resposta: A distância do ponto (3, 4) até a origem é aproximadamente 5.0.
```

```
Ponto p2(-2, 7);
p2.setCoordenadas(1, 1);
double distancia = p2.calcularDistancia();
// Resposta: A distância do ponto (1, 1) até a origem é aproximadamente 1.41421.
```

```
Ponto p3(0, 3);
Ponto p4(4, 0);
double distancia_p3 = p3.calcularDistancia();
double distancia_p4 = p4.calcularDistancia();
// Resposta: A distância do ponto (0, 3) até a origem é 3.0 e do ponto (4, 0) até a origem é 4.0.
```

```
Ponto pontos[3];
pontos[0].setCoordenadas(2, 2);
pontos[1].setCoordenadas(-1, 5);
pontos[2].setCoordenadas(0, 0);

for (int i = 0; i < 3; ++i) {
    double distancia = pontos[i].calcularDistancia();
    cout << "Distância do ponto " << i + 1 << " até a origem: " << distancia << endl;
}
// Resposta: Distância do ponto 1 até a origem: aproximadamente 2.82843
//           Distância do ponto 2 até a origem: aproximadamente 5.09902
//           Distância do ponto 3 até a origem: 0.0
```

```
Ponto p5;
cout << "Coordenadas do ponto p5: (" << p5.getX() << ", " << p5.getY() << ")" << endl;
p5.setCoordenadas(8, -3);
cout << "Novas coordenadas do ponto p5: (" << p5.getX() << ", " << p5.getY() << ")" << endl;
// Resposta: Coordenadas do ponto p5: (0, 0)
//           Novas coordenadas do ponto p5: (8, -3)
```

Parte em Grupos (até 5 pessoas por grupo):

Exercício 3: Sistema de Cadastro de Produtos

Crie um sistema de cadastro de produtos em um supermercado. Cada grupo deve projetar as classes Produto, Estoque e CarrinhoDeCompras. O Produto deve ter atributos como nome, preço e código. O Estoque deve controlar a quantidade de cada produto. O CarrinhoDeCompras deve permitir adicionar e remover produtos, calcular o valor total e exibir o conteúdo.

Testes para o Exercício do Carrinho de Compras:

```
Produto p1("Maçã", 2.5);
Produto p2("Arroz", 10.0);
Produto p3("Leite", 4.0);

CarrinhoDeCompras carrinho;
carrinho.adicionarProduto(p1, 3);
carrinho.adicionarProduto(p2, 2);
carrinho.adicionarProduto(p3, 1);

double valorTotal = carrinho.calcularValorTotal();
cout << "Valor total da compra: " << valorTotal << endl;
// Resposta: Valor total da compra: 31.5
```

```
carrinho.removerProduto(p2, 1);
valorTotal = carrinho.calcularValorTotal();
cout << "Valor total após remoção: " << valorTotal << endl;
// Resposta: Valor total após remoção: 21.5
```

```
carrinho.esvaziarCarrinho();
valorTotal = carrinho.calcularValorTotal();
cout << "Valor total após esvaziar o carrinho: " << valorTotal << endl;
// Resposta: Valor total após esvaziar o carrinho: 0.0
```

```
Produto p4("Chocolate", 3.0);
carrinho.adicionarProduto(p4, 10); // Supondo estoque limitado a 5 unidades
cout << "Quantidade de chocolates no carrinho: " << carrinho.getQuantidadeProduto(p4) << endl;
// Resposta: Quantidade de chocolates no carrinho: 5
```

```
carrinho.adicionarProduto(p1, 2);
carrinho.adicionarProduto(p2, 3);
carrinho.adicionarProduto(p3, 1);
carrinho.adicionarProduto(p4, 2);

carrinho.exibirCarrinho();
// Resposta: Carrinho de Compras:
// - Maçã (2.5) x 2
// - Arroz (10.0) x 3
// - Leite (4.0) x 1
// - Chocolate (3.0) x 2
```

REFERÊNCIAS

Luiz Paulo Moreira Silva. O que é plano cartesiano? Brasil Escola Online. Disponível em: https://brasilescola.uol.com.br/o-que-e/matematica/o-que-e-plano-cartesiano.htm Acessado em 04 de agosto de 2023.

Agostinho Brito. Programação Orientada a Objetos em C++. Artigo online. Disponível em: https://agostinhobritojr.github.io/tutorial/cpp/. Acessado em 04 de agosto de 2023.