EXERCÍCIOS_APOSTILAS 1, 2, 3, 4, 5

1-1- Qual a diferença entre Polimorfismo, Herança, Encapsulamento e Abstração?

Polimorfismo: Capacidade de objetos diferentes responderem de maneira única a uma mesma mensagem.
Herança: Relação entre classes em que uma classe derivada herda características e comportamentos de uma classe base.
Encapsulamento: Proteção de dados e funcionalidades dentro de uma unidade coesa, permitindo acesso controlado por meio de métodos públicos.
Abstração: Foco nos aspectos essenciais de um objeto ou sistema, omitindo detalhes desnecessários e permitindo criar interfaces e classes abstratas para fornecer funcionalidade específica.

1-2- O que é Associação, Agregação, Composição e Generalização?

Associação: Relação entre dois objetos, indicando que um objeto está conectado ao outro de alguma forma.
Agregação: Tipo de associação em que um objeto é composto por outros objetos, mas os objetos agregados podem existir independentemente.
Composição: Tipo de associação mais forte em que um objeto é composto por outros objetos, sendo responsável pelo ciclo de vida desses objetos.
Generalização: Relação de herança entre classes, em que uma classe mais geral (superclasse) é estendida por uma classe mais específica (subclasse), gerando suas características e comportamentos.

2-1- Crie uma classe de nome Carro e atribua a ela todas as propriedades que você acredita que um carro possua. Instancie a classe Carro e preencha 3 objetos distintos. Imprima na tela do usuário todos os atributos dos três carros.

```
class Carro
{
   public string Marca { get; set; }
   public string Modelo { get; set; }
   public int Ano { get; set; }
   public string Cor { get; set; }
}
class Program
{
   static void Main()
   {
      Carro carro1 = new Carro()
      {
         Marca = "Volkswagen",
         Modelo = "Gol Quadrado",
         Ano = 1980,,
         Cor = "Preto"
      };
Carro carro2 = new Carro()
      {
         Marca = "Volkswagen",
         Modelo = "Brasília",
         Ano = 1973,
         Cor = "Amarela"
      };
          Carro carro3 = new Carro()
      {
         Marca = "Volkswagen",
```

```
            Modelo = "Variant",
            Ano = 1969,
            Cor = "Amarela"
        };

        Carro[] carros = { carro1, carro2, carro3 };

        foreach (Carro carro in carros)
        {
            Console.WriteLine("Marca: " + carro.Marca);
            Console.WriteLine("Modelo: " + carro.Modelo);
            Console.WriteLine("Ano: " + carro.Ano);
            Console.WriteLine("Cor: " + carro.Cor);

        }
    }
}
```

2-2- Pesquise como um criar um método específico dentro de uma classe em C#. Crie dois métodos para a classe Carro criada na questão anterior. Um método chamado Acelerar(), que retorna a string "O carro está em movimento" como resposta e outro método Freiar() que também retorna uma string "O carro está parado" como resposta.

```
class Carro
{
    public string Marca { get; set; }
    public string Modelo { get; set; }
    public int Ano { get; set; }
    public string Cor { get; set; }
    public int VelocidadeMaxima { get; set; }
    public void Acelerar()
    {
        Console.WriteLine("O carro está acelerando.");
    }
    public void Frear()
    {
        Console.WriteLine("O carro está freando.");
    }
}
class Program
{
    static void Main()
    {
        Carro carro1 = new Carro()
        {
            Marca = "Volkswagen",
            Modelo = "Gol Quadrado",
            Ano = 1980,
            Cor = "Preto",
            VelocidadeMaxima = 164
        };
        Carro carro2 = new Carro()
        {
            Marca = "Volkswagen",
            Modelo = "Brasília",
            Ano = 1973,
            Cor = "Amarela",
            VelocidadeMaxima = 128,6
```

```
        };
        Carro carro3 = new Carro()
        {
            Marca = "Volkswagen",
            Modelo = "Variant",
            Ano = 1969,
            Cor = "Amarela",
            VelocidadeMaxima = 135
        };
        Console.WriteLine("Carro 1:");
        Console.WriteLine("Marca: " + carro1.Marca);
        Console.WriteLine("Modelo: " + carro1.Modelo);
        Console.WriteLine("Ano: " + carro1.Ano);
        Console.WriteLine("Cor: " + carro1.Cor);
        Console.WriteLine("Velocidade Máxima: " + carro1.VelocidadeMaxima);
        carro1.Acelerar();
        carro1.Frear();
        Console.WriteLine("Carro 2:");
        Console.WriteLine("Marca: " + carro2.Marca);
        Console.WriteLine("Modelo: " + carro2.Modelo);
        Console.WriteLine("Ano: " + carro2.Ano);
        Console.WriteLine("Cor: " + carro2.Cor);
        Console.WriteLine("Velocidade Máxima: " + carro2.VelocidadeMaxima);
        carro2.Acelerar();
        carro2.Frear();
        Console.WriteLine("Carro 3:");
        Console.WriteLine("Marca: " + carro3.Marca);
        Console.WriteLine("Modelo: " + carro3.Modelo);
        Console.WriteLine("Ano: " + carro3.Ano);
        Console.WriteLine("Cor: " + carro3.Cor);
        Console.WriteLine("Velocidade Máxima: " + carro3.VelocidadeMaxima);
        carro3.Acelerar();
        carro3.Frear();
    }
}
```

3-1-Quais são os benefícios de se criar um diagrama de classes?

• Ilustrar modelos de dados para sistemas de informação, não importa quão simples ou complexo seja.
• Entender melhor a visão geral dos esquemas de uma aplicação.
• Expressar visualmente as necessidades específicas de um sistema e divulgar essas informações por toda a empresa.
• Criar gráficos detalhados que destacam qualquer código específico necessário para ser programado e implementado na estrutura descrita.
• Fornecer uma descrição independente de implementação de tipos utilizados em um sistema e passados posteriormente entre seus componentes.

3-2- O que é um array e como é a sua implementação?

Um array é um conjunto de elementos de um mesmo tipo de dados onde cada elemento do conjunto é acessado pela posição no array que é dada através de um índice (uma sequência de números inteiros).

4-1- Faça um programa em C# (com a estrutura do...while) que leia 20 valores inteiros e:
- Encontre e mostre o maior valor;
- Encontre e mostre o menor valor;
- Calcule e mostre a média dos números lidos;

```
int[] valores = new int[6];
int i = 1;
int maiorValor = 0;
int menorValor = 0;
int valor = 0;
int valorMenor = 5;
while (i < 6) {
Console.Write("Infome o " + i+"º valor:  ");
valor = int.Parse(Console.ReadLine());
if (valor > maiorValor){
   valor = maiorValor;
}
if (valor < menorValor){
    menorValor = valor;
}
i++;
}
Console.WriteLine("O maior valor digitado é: " +maiorValor+".");
Console.WriteLine("O menor valor digitado é: " +menorValor+".");
```

4-2- Faça o seguinte programa em C#. Uma loja utiliza o código V para compras à vista e o código P para compras a prazo. Faça um algoritmo que recebe o código (V ou P) e o valor de 15 transações. Calcule e mostre:
- O valor total das compras à vista.
- O valor total das compras a prazo.
- O valor total das compras efetuadas.

```
int CompraaVista = 0;
int CompraaPrazo = 0;
int totalCompras = 0;
for (int i = 1; i <= 15; i++)
{
 Console.WriteLine(" Digite o código da transação ");
    Console.WriteLine("V para à vista, P para à prazo:", i);
    string codigo = Console.ReadLine();
    Console.Write("Digite o valor da transação ", i);
    int valor = int.Parse(Console.ReadLine());
    if (codigo == "V")
    CompraaVista += valor;
    else if (codigo == "P")
    CompraaPrazo += valor;
    totalCompras += valor;
}
Console.WriteLine("Valor total das compras à vista: " + CompraaVista);
Console.WriteLine("Valor total das compras a prazo: " + CompraaPrazo);
Console.WriteLine("Valor total das compras efetuadas: " + totalCompras);
```

4-3- Faça o seguinte programa em C#. A prefeitura de Luziânia fez uma pesquisa com 200 pessoas, coletando dados sobre o salário e o número de filhos. A prefeitura deseja saber:

- A média do salário dessas pessoas.

- A média do número de filhos.

- O maior salário.

- O menor salário.

- A porcentagem de pessoas com salários até R$1500,00

```
    int totalPessoas = 200;
    double somaSalario = 0;
    int somaFilhos = 0;
    double maiorSalario = 0;
    double menorSalario = 0;
    int countSalarioAte1500 = 0;
    for (int i = 1; i <= totalPessoas; i++)
    {
        Console.Write("Digite o salário da pessoa #{0}: R$", i);
        double salario = double.Parse(Console.ReadLine());
        Console.Write("Digite o número de filhos da pessoa #{0}: ", i);
        int numFilhos = int.Parse(Console.ReadLine());
        somaSalario += salario;
        somaFilhos += numFilhos;
        if (salario > maiorSalario)
            maiorSalario = salario;
        if (salario < menorSalario)
            menorSalario = salario;
        if (salario <= 1500)
            countSalarioAte1500++;
    }
    double mediaSalario = somaSalario / totalPessoas;
    double mediaFilhos = (double)somaFilhos / totalPessoas;
    double porcentagemSalarioAte1500 = (double)countSalarioAte1500 / totalPessoas * 100;
    Console.WriteLine("Média do salário: R$" + mediaSalario.ToString("F2"));
    Console.WriteLine("Média do número de filhos: " + mediaFilhos.ToString("F2"));
    Console.WriteLine("Maior salário: R$" + maiorSalario.ToString("F2"));
    Console.WriteLine("Menor salário: R$" + menorSalario.ToString("F2"));
    Console.WriteLine("Porcentagem de pessoas com salários até R$1500,00: " +
porcentagemSalarioAte1500.ToString("F2") + "%");
```

5-1- Faça um programa em C# que leia uma quantidade indefinida de objetos Carro, composto pelos atributos, marca, valor, cor, modelo e ano, e:

- Ordene os carros pelo de maior valor;

- Imprima na tela todos os carros ordenados do maior valor para o de menor valor;

```
class Carro
{
    public string Marca { get; set; }
    public decimal Valor { get; set; }
    public string Cor { get; set; }
    public string Modelo { get; set; }
    public int Ano { get; set; }
}
class Program
{
    static void Main()
```

```
{
List<Carro> carros = new List<Carro>();
while (true)
{
  Carro carro = new Carro();
  Console.Write("Digite a marca do carro (ou digite 'sair' para encerrar): ");
  string marca = Console.ReadLine();
  if (marca.ToLower() == "sair")
     break;
  carro.Marca = marca;
  Console.Write("Digite o valor do carro: ");
  carro.Valor = decimal.Parse(Console.ReadLine());
  Console.Write("Digite a cor do carro: ");
  carro.Cor = Console.ReadLine();
  Console.Write("Digite o modelo do carro: ");
  carro.Modelo = Console.ReadLine();
  Console.Write("Digite o ano do carro: ");
  carro.Ano = int.Parse(Console.ReadLine());
  carros.Add(carro);
  Console.WriteLine("-------------------------");
}
carros.Sort((c1, c2) => c2.Valor.CompareTo(c1.Valor));
Console.WriteLine("Carros ordenados do maior valor para o menor valor:");
foreach (var carro in carros)
{
  Console.WriteLine("Marca: " + carro.Marca);
  Console.WriteLine("Valor: " + carro.Valor);
  Console.WriteLine("Cor: " + carro.Cor);
  Console.WriteLine("Modelo: " + carro.Modelo);
  Console.WriteLine("Ano: " + carro.Ano);

}
}
```

5-2- Baseado no programa anterior (Questão 1) Crie uma interface para Cadastrar, Excluir e Listar os carros.
- Cadastre um carro.
- Exclua um carro.

```
interface ICarroService
{
void CadastrarCarro(List<Carro> carros);
void ExcluirCarro(List<Carro> carros);
void ListarCarros(List<Carro> carros);
}
class Carro : ICarroService
{
public string Marca { get; set; }
public decimal Valor { get; set; }
public string Cor { get; set; }
public string Modelo { get; set; }
public int Ano { get; set; }
public void CadastrarCarro(List<Carro> carros)
{
   Carro novoCarro = new Carro();
```

```
Console.Write("Digite a marca do carro: ");
novoCarro.Marca = Console.ReadLine();
Console.Write("Digite o valor do carro: ");
novoCarro.Valor = decimal.Parse(Console.ReadLine());
Console.Write("Digite a cor do carro: ");
novoCarro.Cor = Console.ReadLine();
Console.Write("Digite o modelo do carro: ");
novoCarro.Modelo = Console.ReadLine();
Console.Write("Digite o ano do carro: ");
novoCarro.Ano = int.Parse(Console.ReadLine());
carros.Add(novoCarro);
Console.WriteLine("Carro cadastrado com sucesso!");
 }
 public void ExcluirCarro(List<Carro> carros)
{
Console.Write("Digite o número do carro que deseja excluir: ");
int numeroCarro = int.Parse(Console.ReadLine());
if (numeroCarro >= 0 && numeroCarro < carros.Count)
{
 carros.RemoveAt(numeroCarro);
 Console.WriteLine("Carro excluído com sucesso!");
 }
else
 {
 Console.WriteLine("Número de carro inválido!");
 }
 }
 public void ListarCarros(List<Carro> carros)
 {
 Console.WriteLine("Lista de carros cadastrados:");
 for (int i = 0; i < carros.Count; i++)
{
Console.WriteLine($"Carro #{i}");
Console.WriteLine("Marca: " + carros[i].Marca);
Console.WriteLine("Valor: " + carros[i].Valor);
Console.WriteLine("Cor: " + carros[i].Cor);
Console.WriteLine("Modelo: " + carros[i].Modelo);
Console.WriteLine("Ano: " + carros[i].Ano);
 }
 }
 }
 class Program
 {
 static void Main()
{
List<Carro> carros = new List<Carro>();
Carro carro = new Carro();
while (true)
{
Console.WriteLine("1 - Cadastrar carro");
Console.WriteLine("2 - Excluir carro");
Console.WriteLine("3 - Listar carros");
Console.WriteLine("0 - Sair");
Console.Write("Escolha uma opção: ");
int opcao = int.Parse(Console.ReadLine());
switch (opcao)
{
case 1:
```

```
carro.CadastrarCarro(carros);
break;
case 2:
carro.ExcluirCarro(carros);
break;
case 3:
carro.ListarCarros(carros);
break;
case 0:
Console.WriteLine("Encerrando o programa...");
return;
default:
Console.WriteLine("Opção inválida! Tente novamente.");
 }
 }
 }
 }
```

5-3- Escreva um programa em C# que deverá ter as seguintes opções:
- Carregar Vetor.
- Listar Vetor.
- Exibir apenas os números pares do vetor.
- Exibir apenas os números ímpares do vetor.
- Exibir a quantidade de números pares existentes nas posições ímpares do vetor. - Exibir a quantidade de números ímpares existentes nas posições pares do vetor. - Sair

```
class Program
{
static int[] vetor;
static int tamanho;
static void Main()
{
Console.WriteLine("Programa do Vetor");
while (true)
{
Console.WriteLine("1 - Carregar Vetor");
Console.WriteLine("2 - Listar Vetor");
Console.WriteLine("3 - Exibir números pares do vetor");
Console.WriteLine("4 - Exibir números ímpares do vetor");
Console.WriteLine("5 - Quantidade de pares em posições ímpares");
Console.WriteLine("6 - Quantidade de ímpares em posições pares");
Console.WriteLine("0 - Sair");
Console.Write("Escolha uma opção: ");
int opcao = int.Parse(Console.ReadLine())
switch (opcao)
{
case 1:
CarregarVetor();
Console.WriteLine("Vetor carregado com sucesso!")
break;
case 2:
ListarVetor();
break;
case 3:
```

```
            ExibirPares();
            break;
            case 4:
            ExibirImpares();
            break;
            case 5:
            QuantidadeParesPosicoesImpares();
            break;
            case 6:
            QuantidadeImparesPosicoesPares();
            break;
            case 0:
            Console.WriteLine("Encerrando o programa...");
            return;
            default:
            Console.WriteLine("Opção inválida! Tente novamente.");
            break;
        }
    }
}
static void CarregarVetor()
{
Console.Write("Digite o tamanho do vetor: ");
tamanho = int.Parse(Console.ReadLine());
vetor = new int[tamanho]:
for (int i = 0; i < tamanho; i++)
{
Console.Write("Digite o valor para a posição {0}: ", i);
vetor[i] = int.Parse(Console.ReadLine());
}
}
static void ListarVetor()
{
Console.WriteLine("Vetor:");
for (int i = 0; i < tamanho; i++)
{
Console.WriteLine("Posição {0}: {1}", i, vetor[i]);
}
}
static void ExibirPares()
{
Console.WriteLine("Números pares do vetor:"):
for (int i = 0; i < tamanho; i++)
{
if (vetor[i] % 2 == 0)
{
Console.WriteLine(vetor[i]);
}
}
}
static void ExibirImpares()
{
Console.WriteLine("Números ímpares do vetor:");
for (int i = 0; i < tamanho; i++)
{
if (vetor[i] % 2 != 0)
{
Console.WriteLine(vetor[i]);
```

```
}
}
}
static void QuantidadeParesPosicoesImpares()
{
int count = 0;
for (int i = 1; i < tamanho; i += 2)
{
if (vetor[i] % 2 == 0)
{
count++;
}
}
Console.WriteLine("Quantidade de números pares em posições ímpares: {0}", count);
}
```