­­­PONTIFÍCIA UNIVERSIDADE cATÓLICA DE MINAS GERAIS

Graduação em Sistemas de Informação

**Andrew de Oliveira Duchini**

**Cristóvão Olegário de Castro­­­**

Relatório 3

Laboratório de Programação Orientada por Objeto - Professor: Paulo Amaral

Belo Horizonte

2015

**Exercício 1.2**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Ex1\_2

{

class Program

{

public class PFisica : Contribuinte

{

protected string cpf;

protected double salario;

public PFisica(string n, string end, double sal, String c)

{

this.nome = n;

this.endereco = end;

cpf = c;

salario = sal;

Console.WriteLine(nome + " imposto de " + calcImposto());

}

public override double calcImposto()

{

double imposto = 0;

if (salario > 0 && salario <= 1400)

{ }

else if (salario > 1400 && salario <= 2100)

{

imposto = salario \* 0.1;//10% de imposto

}

else if (salario > 2100 && salario <= 2800)

{

imposto = salario \* 0.15;//15% de imposto

}

else if (salario > 2800 && salario <= 3600)

{

imposto = salario \* 0.25;//20% de imposto

}

else if (salario > 3600)

{

imposto = salario \* 0.30;//30% de imposto

}

else

{

Console.WriteLine("salario inválido");

}

return (imposto);

}

}

public class PJuridica : Contribuinte

{

protected String cnpj;

protected double faturamento;

public PJuridica(String n, string end, double f, String c)

{

this.nome = n;

this.endereco = end;

faturamento = f;

cnpj = c;

Console.WriteLine(nome + " imposto de " + calcImposto());

}

public override double calcImposto()

{

return (faturamento \* 0.1);

}

}

public abstract class Contribuinte

{

protected String nome;

protected String endereco;

public static Contribuinte[] listaContr()

{

Contribuinte[] lst = new Contribuinte[6];

lst[0] = new PFisica("Joao Santos", "Rua abc, 123", 3000.00, "11111");

lst[1] = new PJuridica("Lojas AA", "Rua Hum, 111", 150000.00, "10055");

lst[2] = new PFisica("Maria Soares", "Av. Xyz, 777", 5000.00, "22222");

lst[3] = new PJuridica("Supermercados B", "Rua Dois, 987", 2000000.00, "10066");

lst[4] = new PFisica("Carla Maia", "Av. Três, 333", 1500.00, "33333");

lst[5] = new PJuridica("Posto XX", "Rua Cinco, 555", 500000.00, "10077");

return lst;

}

public String getNome()

{

return nome;

}

abstract public double calcImposto();

}

static void Main(string[] args)

{

Console.WriteLine("Programadores: Andrew de Oliveira Duchini\n\tCristóvão Olegário\n\n");

Contribuinte.listaContr();

Console.ReadKey();

}

}

}

**Exercício 2.1**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Ex2\_1

{

class Program

{

static void Main(String[] args)

{

Console.WriteLine("Programadores: Andrew de Oliveira Duchini\n\tCristóvão Olegário\n\n");

TV tv = new TV();

tv.ligar();

tv.aumentar\_volume();

tv.diminuir\_volume();

tv.desligar();

Console.ReadKey();

}

public interface Icontrole\_remoto

{

void ligar();

void desligar();

}

public class TV : Icontrole\_remoto

{

int volume, canal;

public TV()

{

volume = 100;

canal = 1;

}

public void aumentar\_volume()

{

if (volume < 100)

{

volume++;

}

else

{

Console.WriteLine("volume máximo");

}

}

public void diminuir\_volume()

{

if (volume > 1)

{

volume--;

}

else

{

Console.WriteLine("volume mínimo");

}

}

public void ligar()

{

Console.WriteLine("TV ligada");

}

public void desligar()

{

Console.WriteLine("TV desligada");

}

public void subir\_canal()

{

if (canal < 83)

{

canal++;

}

else

{

Console.WriteLine("Último canal");

}

}

public void descer\_canal()

{

if (canal > 1)

{

canal--;

}

else

{

Console.WriteLine("Canal mínimo");

}

}

}

public class DVD : Icontrole\_remoto

{

public void ligar()

{ }

public void desligar()

{ }

}

}

}

**Exercício 2.2**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Ex2\_2

{

class Program

{

static void Main(String[] args)

{ Console.WriteLine("Programadores: Andrew de Oliveira Duchini\n\tCristóvão Olegário\n\n");

int num\_formas = 0;

string tipo\_de\_forma = "";

Console.WriteLine("Quantas formas deseja criar?");

try

{

num\_formas = Convert.ToInt32(Console.ReadLine());

}

catch (FormatException)

{

Console.WriteLine("Formato errado");

}

catch (NotFiniteNumberException)

{

Console.WriteLine("inválido");

}

IFormas[] vet = new IFormas[num\_formas];

for (int i = 0; i < num\_formas; i++)

{

Console.WriteLine("Deseja criar um círculo [c], quadrado[q] ou retangulo[r]?");

tipo\_de\_forma = Console.ReadLine();

switch (tipo\_de\_forma)

{

case "c":

int r;

Console.WriteLine("Digite o valor do raio");

r = Convert.ToInt32(Console.ReadLine());

vet[i] = new circulo(r);

break;

case "q":

int l;

Console.WriteLine("Digite o lado do quadrado");

l = Convert.ToInt32(Console.ReadLine());

vet[i] = new quadrado(l);

break;

case "r":

int l1, l2;

Console.WriteLine("Digite o primeiro lado do retangulo");

l1 = Convert.ToInt32(Console.ReadLine());

Console.WriteLine("Digite o outro lado");

l2 = Convert.ToInt32(Console.ReadLine());

vet[i] = new retangulos(l1, l2);

break;

}

}

for (int j = 0; j < vet.Length; j++)

{

vet[j].imprime();

Console.WriteLine();

}

Console.ReadKey();

}

public interface IFormas

{

double cal\_Perimetro();

double cal\_area();

void imprime();

}

public class retangulos : IFormas

{

int l1, l2;

public retangulos(int l1, int l2)

{

this.l1 = l1;

this.l2 = l2;

}

public double cal\_area()

{

return (l1 \* l2);

}

public double cal\_Perimetro()

{

return ((2 \* l1) + (2 \* l2));

}

public void imprime()

{ Console.WriteLine("Tipo de Forma : Retângulo\nTamanho dos lados : " + l1 + " e " + l2 + "\nPerímetro : " + cal\_Perimetro() + "\nÁrea : " + cal\_area() + "");

}

}

public class quadrado : IFormas

{

int lado;

public quadrado(int l)

{

this.lado = l;

}

public double cal\_area()

{

return (Math.Pow(lado, 2));

}

public double cal\_Perimetro()

{

return (lado \* 4);

}

public void imprime()

{ Console.WriteLine("Tipo de Forma : Quadrado\nTamanho dos lado : " + lado + "\nPerímetro : " + cal\_Perimetro() + "\nÁrea : " + cal\_area() + "");

}

}

public class circulo : IFormas

{

int raio;

public circulo(int raio)

{

this.raio = raio;

}

public double cal\_area()

{

return (Math.PI \* (Math.Pow(raio, 2)));

}

public double cal\_Perimetro()

{

return (2 \* Math.PI \* raio);

}

public void imprime()

{ Console.WriteLine("Tipo de Forma : Círculo\nTamanho dos lado : " + raio + "\nPerímetro : " + cal\_Perimetro() + "\nÁrea : " + cal\_area() + "");

}

}

}

}

**Exercício 3.1**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Ex3\_1

{

class Program

{

class Funcionario\_Generico

{

protected string nome;

protected int cpf;

protected double salario;

}

class Gerente : Funcionario\_Generico

{

Controle\_de\_Ponto c;

public Gerente(string nome, int cpf, double salario)

{

this.nome = nome;

this.cpf = cpf;

this.salario = salario;

}

public void horas()

{

c = new Controle\_de\_Ponto();

}

}

class Telefonista : Funcionario\_Generico

{

Controle\_de\_Ponto c;

public Telefonista(string nome, int cpf, double salario)

{

this.nome = nome;

this.cpf = cpf;

this.salario = salario;

}

public void horas()

{

c = new Controle\_de\_Ponto();

}

}

class Controle\_de\_Ponto

{

int[] hora\_de\_entrada = new int[2];

int[] hora\_de\_saida = new int[2];

public Controle\_de\_Ponto()

{

string hora;

Console.WriteLine("digite o horario de entrada, separado por ':'");

hora = Console.ReadLine();

string[] hora\_de\_entrada = (hora.Split(':'));

Console.WriteLine("digite o horario de saida, separado por ':'");

hora = Console.ReadLine();

string[] hora\_de\_saida = (hora.Split(':'));

}

}

static void Main(String[] args)

{

Console.WriteLine("Programadores: Andrew de Oliveira Duchini\n\tCristóvão Olegário");

Telefonista t = new Telefonista("Joana", 1233432, 1000);

Gerente g = new Gerente("Geraldo", 5432345, 2000);

g.horas();

t.horas();

}

}

}

**Exercício 3.2**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Ex3\_2

{

class Program

{

static void Main(String[] args)

{ Console.WriteLine("Programadores: Andrew de Oliveira Duchini\n\tCristóvão Olegário");

//TESTE

ContaPoupanca a = new ContaPoupanca(500.0, 16);

ContaCorrente b = new ContaCorrente(1000, 700);

GeradorDeExtrato c = new GeradorDeExtrato();

c.ImprimeExtratoBasico(a);

c.ImprimeExtratoBasico(b);

Console.ReadKey();

}

class Conta

{

public double Saldo { set; get ; }

}

class ContaPoupanca : Conta

{

public int DiaDoAniversario { get ; set ; }

public ContaPoupanca(double saldo, int dia)

{

Saldo = saldo;

DiaDoAniversario = dia;

}

}

class ContaCorrente : Conta

{

public double Limite { get ; set; }

public ContaCorrente(double saldo, double limite)

{

Saldo = saldo;

Limite = limite;

}

}

class GeradorDeExtrato

{

public void ImprimeExtratoBasico(Conta c)

{

DateTime agora = DateTime.Now;

string horario = String.Format(" {0: d/M/ yyyy HH:mm:ss}", agora);

Console.WriteLine(" DATA : " + horario);

Console.WriteLine(" SALDO : " + c.Saldo);

}

}

}

}

**Exercicio 3.3**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Ex3\_3

{

class Program

{

interface IConta

{

string nome\_do\_titular{get;set;}

int num\_conta { get; set; }

void Deposita(double valor);

void Saca(double valor);

double Saldo { get; set; }

void preenche(string titular, int num);

}

class ContaCorrente : IConta

{

public double Saldo { get; set; }

public string nome\_do\_titular { get; set; }

public int num\_conta { get; set; }

private double taxaPorOperacao = 0.45;

public void Deposita(double valor)

{

this.Saldo += valor - this.taxaPorOperacao;

}

public void Saca(double valor)

{

if (valor < this.Saldo)

{

this.Saldo -= valor + this.taxaPorOperacao;

}

else

{

Console.WriteLine("operação inváida");

}

}

public void preenche(string titular, int num)

{

this.nome\_do\_titular = titular;

this.num\_conta = 100+num;

}

}

class ContaPoupanca : IConta

{

public double Saldo { get; set; }

public string nome\_do\_titular { get; set; }

public int num\_conta { get; set; }

public void Deposita(double valor)

{

this.Saldo += valor;

}

public void Saca(double valor)

{

if (valor < this.Saldo)

{

this.Saldo -= valor;

}

else

{

Console.WriteLine("operação inváida");

}

}

public void preenche(string titular, int num)

{

this.nome\_do\_titular = titular;

this.num\_conta = 200+num;

}

}

class GeradorDeExtrato

{

public void GeraExtrato(IConta c)

{

System.Console.WriteLine(" EXTRATO ");

System.Console.WriteLine(" SALDO : " + c.Saldo);

}

}

public void percorreVetor() { }

static void Main(String[] args)

{ Console.WriteLine("Programadores: Andrew de Oliveira Duchini\n\tCristóvão Olegário\n\n");

IConta[] vet = new IConta[100];//armazenamento das contas.

int indice = 0, i = 0;

string tipo\_de\_conta;

while (indice != 7)

{

Console.WriteLine("[1] - Criar uma nova conta");

Console.WriteLine("[2] - Excluir uma conta");

Console.WriteLine("[3] - Depositar em uma conta");

Console.WriteLine("[4] - Sacar de uma conta");

Console.WriteLine("[5] - Imprimir o extrato de uma conta");

Console.WriteLine("[6] - Imprimir uma relação das contas existentes");

Console.WriteLine("[7] - Sair do programa");

indice = Convert.ToInt32(Console.ReadLine());

switch (indice)

{

case 1:

string nome;

Console.WriteLine("Digite o tipo de conta");

tipo\_de\_conta = Console.ReadLine();

Console.WriteLine("Digite seu nome");

nome = Console.ReadLine();

if (tipo\_de\_conta == "corrente")

{

vet[i] = new ContaCorrente();

vet[i].preenche(nome, i);

Console.WriteLine("O número da sua conta é: " + vet[i].num\_conta);

}

else if (tipo\_de\_conta == "poupanca")

{

vet[i] = new ContaPoupanca();

vet[i].preenche(nome, i);

Console.WriteLine("O número da sua conta é: " + vet[i].num\_conta);

}

i++;

break;

case 2:

Console.WriteLine("Digite o numero da conta a ser excluida");

int num = Convert.ToInt32(Console.ReadLine());

int g;

for ( g = 0; g < vet.Length; g++)

{

try

{

if (num == vet[g].num\_conta)

{

vet[g] = null;

g = vet.Length;

Console.WriteLine("Conta excluída");

}

}

catch (NullReferenceException)

{ }

}

if(g == 100)

{

Console.WriteLine("Conta nao encontrada");

}

break;

case 3:

Console.WriteLine("Digite o número da conta");

int nume = Convert.ToInt32(Console.ReadLine());

for (g = 0; g < vet.Length; g++)

{

try

{

if (nume == vet[g].num\_conta)

{

Console.WriteLine("Conta encontrada");

Console.WriteLine("Digite o valor a ser depositado");

double deposito = Convert.ToDouble(Console.ReadLine());

vet[g].Deposita(deposito);

g = vet.Length;

}

}

catch (NullReferenceException) { }

}

if (g == 100)

{

Console.WriteLine("Conta nao encontrada");

}

break;

case 4:

Console.WriteLine("Digite o numero da conta a ser sacada");

nume = Convert.ToInt32(Console.ReadLine());

for (g = 0; g < vet.Length; g++)

{

try

{

if (nume == vet[g].num\_conta)

{

Console.WriteLine("Conta encontrada");

Console.WriteLine("Digite o valor a ser sacado");

double saque = Convert.ToDouble(Console.ReadLine());

vet[g].Saca(saque);

g = vet.Length;

}

}

catch (NullReferenceException) { }

}

if (g == 100)

{

Console.WriteLine("Conta nao encontrada");

}

break;

case 5:

Console.WriteLine("Digite o numero da conta");

nume = Convert.ToInt32(Console.ReadLine());

for (g = 0; g < vet.Length; g++)

{

try

{

if (nume == vet[g].num\_conta)

{

Console.WriteLine("Conta encontrada");

Console.WriteLine("Seu saldo é de : R$ " + vet[g].Saldo);

g = vet.Length;

}

}

catch (NullReferenceException) { }

}

if (g == 100) { Console.WriteLine("Conta nao encontrada"); }

break;

case 6:

int aux = 0;

Console.WriteLine("Relação das contas existentes :");

for (g = 0; g < vet.Length; g++)

{

if (vet[g] != null)

{ Console.WriteLine("\nNome : " + vet[g].nome\_do\_titular + "\nNúmero da conta : " + vet[g].num\_conta);

}

else

{

aux++;

}

}

if (aux == 100)

{

Console.WriteLine("\nNão há contas para serem exibidas\n");

}

break;

case 7:

break;

}

}

}

}

}

**Exercício 3.4**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Ex3\_4

{

class Telefone

{

protected string TipodoTelefone = " ";

public virtual void Ring()

{

Console.WriteLine("Tocando o {0}", TipodoTelefone);

}

public Telefone()

{

TipodoTelefone = "";

}

class TelefoneEletronico : Telefone

{

public TelefoneEletronico()

{

set("digital");

}

public void set(string set)

{

TipodoTelefone = set;

}

public override void Ring()

{

Console.WriteLine("Mensagem Diferente");

}

}

static void Main(string[] args)

{ Console.WriteLine("Programadores: Andrew de Oliveira Duchini\n\tCristóvão Olegário");

TelefoneEletronico t = new TelefoneEletronico();

t.Ring();

Console.ReadKey();

}

}

}