Lista de exercícios 1

TI224 - Programação Multiplataforma

Primeiro Semestre de 2024

Guilherme Macedo

Exercício 1

Qual o retorno deste programa para `funcao([83, 41, 5, 1, 59, 97], 2)`?

```
1     public static int[] funcao(int[] A, int i)
2     {
3         A[1] = 17;
4         A[i / 2] = 9;
5         A[2 * i - 1] = 95;
6         A[i - 1] = A[5] / 2;
7         A[3] = A[i];
8         A[i + 1] = A[i] + A[i - 1];
9         A[A[2] - 2] = 78;
10        A[A[i] - 1] = A[1] * A[i] / 5;
11        A[A[2] % 2 + 2] = A[i + 6 / 2] - A[i - 1 * 2];
12        return A;
13    }
```

Exercício 2

Qual o retorno deste programa para `funcao(81)`?

```
1     public static int funcao(int n)
2     {
3         int p = 1, r = n;
4         while (p + 1 < r)
5         {
6             int q = (p + r) / 2;
7             if (Math.Pow(q, 2) <= n)
8                 p = q;
9             else
10                r = q;
11        }
12        return p;
13    }
```

Exercício 3

O programa abaixo recebe um arranjo A de n números inteiros e o rearranja de modo que seus elementos, ao final, estejam ordenados de forma decrescente. Contudo, este programa possui alguns erros de lógica. Encontre-os e corrija-os.

```
1     public static int[] ordena(int[] A)
2     {
3         int i, j, chave;
4         for (i = 1; i < A.Length; i++)
5         {
6             chave = A[i];
7             j = i - 1;
8             while (j >= 0 && A[j] > chave) {
9                 A[j + 1] = A[j];
10                j = j - 1;
11            }
```

```
12              A[j + 1] = chave;
13          }
14          return A;
15      }
```

Exercício 4

Observe o programa abaixo e responda:

```
1       private static int busca1(int[] A, int k)
2       {
3           for (int i = 0; i < A.Length; i++)
4               if (A[i] == k)
5                   return i;
6           return -1;
7       }
8
9       private static int busca2(int[] A, int k)
10      {
11          int inicio = 0, fim = A.Length;
12          while (inicio < fim)
13          {
14              int meio = (inicio + fim) / 2;
15              if (A[meio] == k)
16                  return meio;
17              if (A[meio] < k)
18                  inicio = meio + 1;
19              else
20                  fim = meio - 1;
21          }
22          return -1;
```

```
23      }
24
25      public static void Main(string[] args)
26      {
27          int[] A = {1, 3, 5, 7, 9, 11, 13, 15, 17, 19};
28          int valor1 = busca1(A, 13);
29          int valor2 = busca2(A, 13);
30          Console.WriteLine(valor1 + " - " + valor2);
31      }
```

(a) Qual é o resultado da impressão da linha 30?

(b) `busca1` é "melhor" do que `busca2`? Justifique sua resposta.

Exercício 5

Escreva um programa em `C#` que embaralhe um arranjo A de n inteiros.

Exercício 6

Escreva um programa em `C#` que encontre dois elementos de um arranjo A de n inteiros, distintos entre si, que somados seja igual a um determinado inteiro k.

Exercício 7

Escreva um programa em `C#` que remova os elementos duplicados de um arranjo A de n cadeias de caracteres.

Exercício 8

Escreva um programa em `C#` que organize um arranjo A de n inteiros de modo que todos os inteiros negativos aparaçam antes de todos os inteiros positivos.

Exercício 9

Escreva um programa em `C#` que obtenha o elemento majoritário de um arranjo A de n inteiros. Um elemento majoritário é um elemento que aparece mais de $n/2$ vezes.

Exercício 10

Escreva um programa em `C#` que encontre um elemento específico em um arranjo A de n inteiros usando a busca por interpolação.

Exercício 11

Escreva um programa em `C#` que represente o digrama de classes da Figura 1.

Exercício 13

Escreva um programa em `C#` que represente o digrama de classes da Figura 2.

Exercício 14

Escreva um programa em `C#` que represente o digrama de classes da Figura 3.

Exercício 15

Escreva um programa em `C#` que represente o digrama de classes da Figura 4. Suponha que um pedido não tenha mais de 30 produtos ao mesmo tempo.

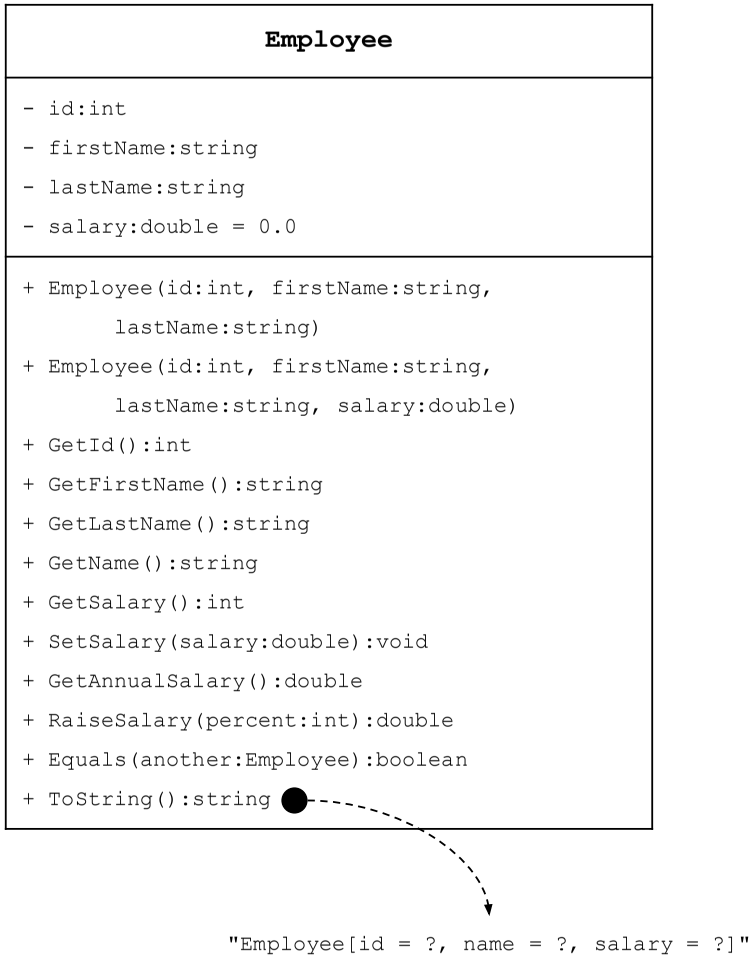
Employee
- id:int
- firstName:string
- lastName:string
- salary:double = 0.0
+ Employee(id:int, firstName:string,
lastName:string)
+ Employee(id:int, firstName:string,
lastName:string, salary:double)
+ GetId():int
+ GetFirstName():string
+ GetLastName():string
+ GetName():string
+ GetSalary():int
+ SetSalary(salary:double):void
+ GetAnnualSalary():double
+ RaiseSalary(percent:int):double
+ Equals(another:Employee):boolean
+ ToString():string
"Employee[id = ?, name = ?, salary = ?]"

Figura 1

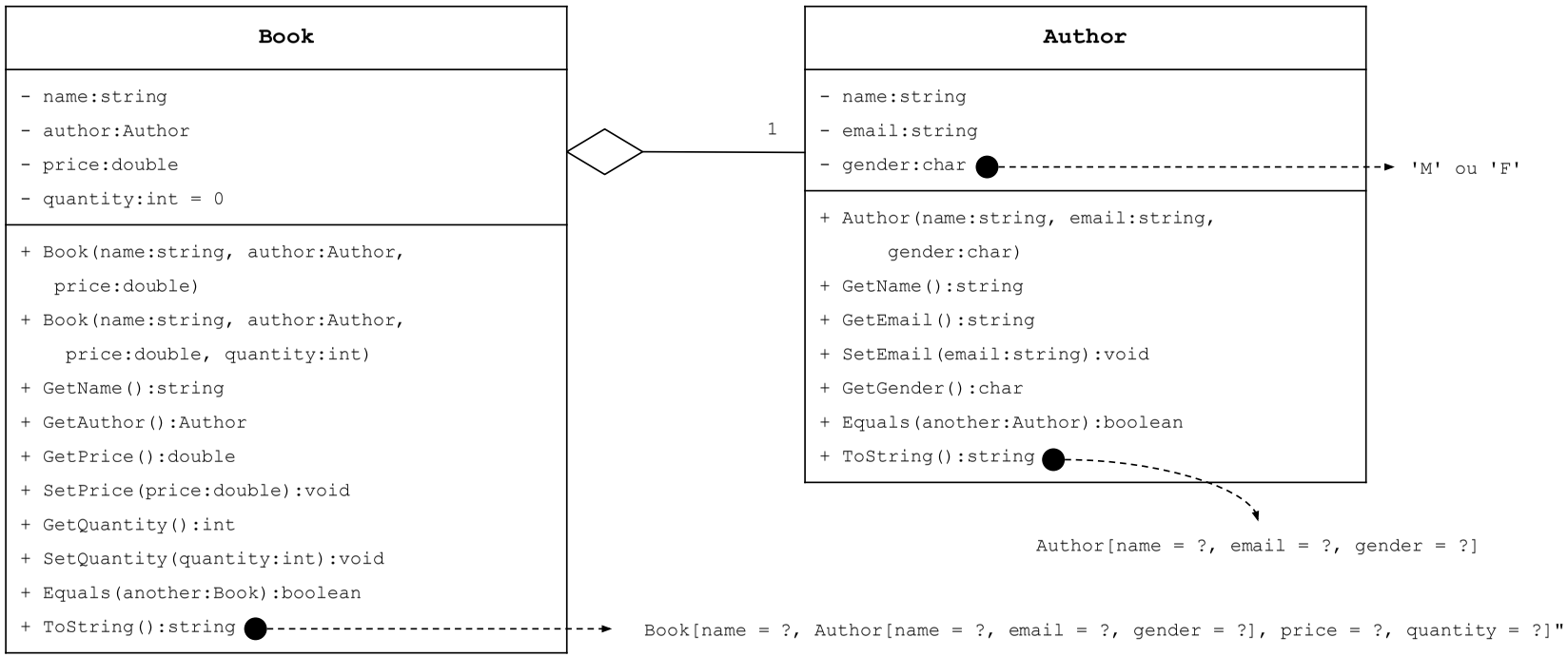
Book
- name:string
- author:Author
- price:double
- quantity:int = 0
+ Book(name:string, author:Author, price:double)
+ Book(name:string, author:Author, price:double, quantity:int)
+ GetName():string
+ GetAuthor():Author
+ GetPrice():double
+ SetPrice(price:double):void
+ GetQuantity():int
+ SetQuantity(quantity:int):void
+ Equals(another:Book):boolean
+ ToString():string
Book[name = ?, Author[name = ?, email = ?, gender = ?], price = ?, quantity = ?]"
1
Author
- name:string
- email:string
- gender:char
'M' ou 'F'
+ Author(name:string, email:string, gender:char)
+ GetName():string
+ GetEmail():string
+ SetEmail(email:string):void
+ GetGender():char
+ Equals(another:Author):boolean
+ ToString():string
Author[name = ?, email = ?, gender = ?]

Figura 2

```
MoveUp:    y += speed
MoveDown:  y -= speed
MoveLeft:  x -= speed
MoveRight: x += speed
```

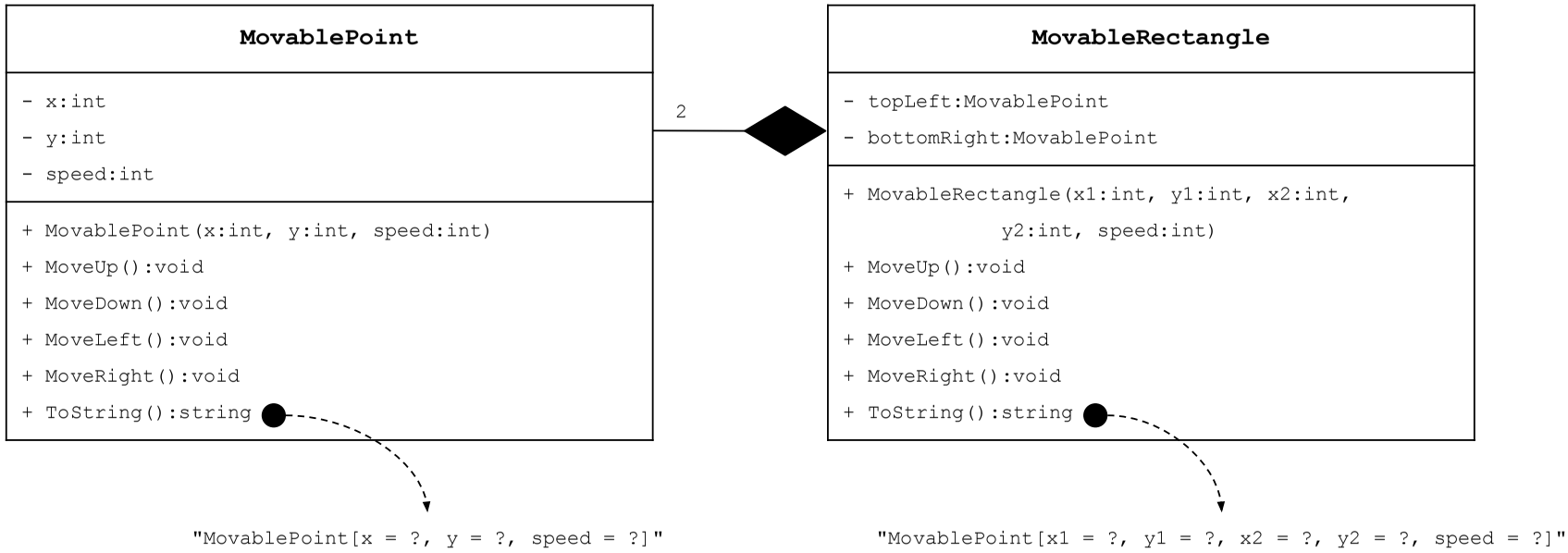

Figura 3

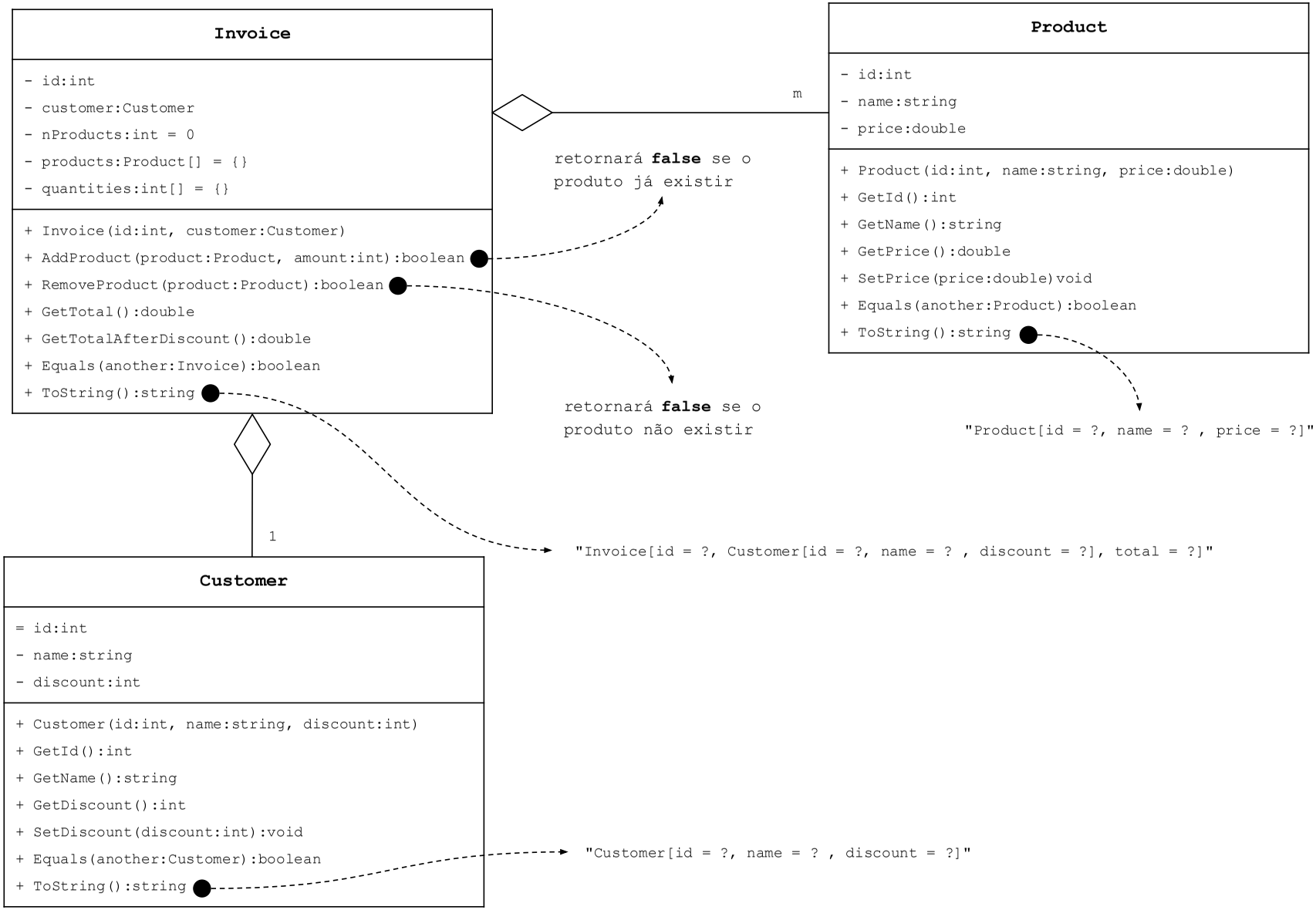
Invoice
- id:int
- customer:Customer
- nProducts:int = 0
- products:Product[] = {}
- quantities:int[] = {}
+ Invoice(id:int, customer:Customer)
+ AddProduct(product:Product, amount:int):boolean
+ RemoveProduct(product:Product):boolean
+ GetTotal():double
+ GetTotalAfterDiscount():double
+ Equals(another:Invoice):boolean
+ ToString():string
m
Product
- id:int
- name:string
- price:double
+ Product(id:int, name:string, price:double)
+ GetId():int
+ GetName():string
+ GetPrice():double
+ SetPrice(price:double)void
+ Equals(another:Product):boolean
+ ToString():string
retornará false se o produto já existir
retornará false se o produto não existir
"Product[id = ?, name = ? , price = ?]"
1
"Invoice[id = ?, Customer[id = ?, name = ? , discount = ?], total = ?]"
Customer
= id:int
- name:string
- discount:int
+ Customer(id:int, name:string, discount:int)
+ GetId():int
+ GetName():string
+ GetDiscount():int
+ SetDiscount(discount:int):void
+ Equals(another:Customer):boolean
+ ToString():string
"Customer[id = ?, name = ? , discount = ?]"

Figura 4