

RE
COD
ME_

RE qualifica-te |

Ficha de avaliação de capacidades

Programação C# (Parte 1)



Rua Flores de Lima N°16, Lisboa
cv@recodme.pt



INSTITUTO DO EMPREGO E FORMAÇÃO PROFISSIONAL

Lisb@20²⁰

PORTUGAL
2020

UNIÃO EUROPEIA
Fundo Social Europeu

Regras

Esta ficha tem como intuito avaliar o teu conhecimento e capacidades acerca das bases de programação em C#. Deverás, até as 12h50 de hoje, entregar a sua resolução por email, ou até por repositório até ao final, ou não será contabilizada. Durante este período, poderás consultar os materiais lecionados (slides), tendo sempre em consideração o tempo que te resta até à entrega. Se sentires que consegues escrever mais depressa em papel, poderás então escrever as tuas respostas numa folha, tirar foto ou digitalizar, enviando juntamente com a solução. Caso tenhas algum problema notifica-o no Slack. Um dos teus colegas poderá estar na mesma situação! Assim que entregares, notifica-me.

Atenção : o email enviado com a solução para a ficha deverá conter apenas um anexo (zip) com o nome FICHA_CSHARP_PrimeiroNome_UltimoNome (ex: FICHA_CSHARP_Fabio_Jesus.zip)

A ficha é composta por 4 (quatro) grupos:

I. Verdadeiros e Falsos (50 pontos)

As respostas deverão ser colocadas nos respetivos campos da grelha.

II. Desenvolvimento (60 pontos)

A resposta deve ser colocada abaixo da pergunta, ou se escreveres a resposta numa folha, marca apenas o número da questão. (ex: 1))

III. Prático (90 pontos)

As respostas devem ser colocadas no código fonte. Caso ocorra algum erro que cause o teu projeto a funcionar, respira fundo, comenta o código e continua. Não deixes que um erro mínimo te impeça de continuar a ficha, pois todo o código comentado será avaliado, e caso esteja parcialmente correto, será atribuída essa pontuação. Perguntas que tenham cotações diferentes apenas totalizam a pontuação total se forem apresentadas soluções para cada uma (ex: [iterativa 5pts / recursiva 10pts] resulta em 15 pontos se entregares ambas).

IV. Extras (50 pontos)

São pontos extra, por isso, tal como nas outras fichas e testes, nunca contam para além de demonstrar o teu esforço. Aconselho-te a resolver estes exercícios assim que acabares a ficha.

Boa sorte!

Grupo I – Verdadeiros e Falsos (50 pontos)

1	2	3	4	5	6	7	8	9	10

1. A framework une ferramentas tais como o editor e o compilador para facilitar o desenvolvimento de software.
2. As linguagens de programação são classificadas como DSL ou GPL.
3. A leitura do conteúdo introduzido pelo utilizador pode ser realizada através da função `Console.ReadLine();`
4. A negação de uma proposição é realizada através do operador ? (ponto de interrogação)
5. "1nome" é um nome válido para uma variável
6. O tratamento de exceções permite corrigir problemas causados pelo programador.
7. Após terminar a execução de um âmbito, todas as variáveis declaradas no seu interior ficam inacessíveis.
8. A arquitetura .NET Framework é composta por Sistema Operativo, Modelos de aplicação, FCL e CLR.
9. O *debugger* é útil para, durante em tempo de execução, detetar anomalias.
10. De momento, o Visual Studio é apenas uma janela, onde podem ser instalados vários plugins que podem ser ancorados para uso posterior.

Grupo II – Desenvolvimento (60 pontos)

1. Distingue as DSL de GPL. [6 pontos]
2. Descreve as nove ferramentas de um IDE [9 pontos]
3. Identifica as sete características do C# [7 pontos]
4. Distingue IDE de Framework [6 pontos]
5. Descreve boxing, unboxing e casting. Apresenta exemplos para cada processo. [6 pontos]
6. Descreve as formas de juntar texto com variáveis, apresentando exemplos. [6 pontos]
7. Identifica o que pode ser realizado com a .NET Framework. [6 pontos]
8. Descreve a utilidade de um *breakpoint*. Apresenta um exemplo. [6 pontos]
9. Descreve a hierarquia de uma aplicação [8 pontos]

Grupo III – Desenvolvimento (90 pontos)

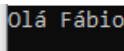
1. Na solução em anexo com o enunciado, criar uma biblioteca chamada Grupo3. Nesta biblioteca deverá ser criada uma função chamada GoodbyeWorld que apresente, na consola, “Good bye world!”. [10 pontos]

```
static void Main(string[] args)
{
    Console.WriteLine("Hello World!");
    Grupo3.Gruop3.GoodbyeWorld();
}
```



2. Na biblioteca Grupolll, no ficheiro Exercicio2.cs:
- a. Cria uma função que solicite o nome do utilizador, e apresente “Olá” seguido do nome [4 pontos]

```
//III-2
Exercicio1.OlaPessoa("Fábio");
```



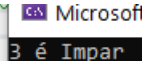
- b. Na função “QuantasPatas”, implementa o algoritmo necessário para devolver o número de patas de um conjunto de vacas, porcos e galinhas. [4 pontos]

```
Console.WriteLine(Exercicio1.QuantasPatas(3, 4, 2));
```




- c. Cria uma região de código [2 ponto]
- d. Cria uma função que verifique se um número introduzido é par ou ímpar. Se criaste uma região, coloca a função dentro da região [4 pontos]

```
Exercicio2.ParOuImpar(3);
```



- e. Cria uma função que calcule a soma de dois valores inteiros. Caso estes sejam iguais, calcula o triplo da soma. [4 pontos]

```
Exercicio2.Somar(1, 2);
Exercicio2.Somar(1, 1);
```



- f. Cria uma função que apresente a potência de um número. Deverá requisitar o número e a potência. [4 pontos]

```
Console.WriteLine(Exercicio2.Potencia(2,3));
```



3. Na biblioteca Grupolll, no ficheiro Exercicio3.cs:

- a. Cria uma função que solicite ao utilizador dois números inteiros (m e n) e um caracter, e que apresente uma tabela mxn preenchida com o caracter. **[10 pontos]**

```
Exercicio3.ApresentarTabela(); Quantas linhas?  
3  
Quantas colunas?  
2  
Qual o caracter?  
a  
a a  
a a  
a a
```

- b. Cria uma função que verifica se um número é múltiplo de 3 e/ou 7. **[10 pontos]**

```
Exercicio3.MultiploDeTresESete(21); Multiplo de 3 e 7  
Exercicio3.MultiploDeTresESete(18); Multiplo de 3  
Exercicio3.MultiploDeTresESete(14); Multiplo de 7  
Exercicio3.MultiploDeTresESete(16); Nem multiplo de 3 nem de 7
```

- c. Cria uma função que apresente o fatorial de um número. **[5 pontos iterativa / 8 recursiva]**

```
Exercicio3.Fatorial(); Qual o número?  
3  
3! = 3 x 2 x 1 = 6
```

4. Na biblioteca Grupolll, no ficheiro Exercicio4.cs:

- a. Cria uma função que solicite dois números, a e b, e um caracter op. De acordo com 'op' deverão realizar o conjunto de operações de soma, subtração multiplicação, divisão e resto de divisão e apresentar o seu resultado. **[10 pontos]**

```
Exercicio4.Calculadora(); Primeiro Número?  
4  
Segundo Número?  
7  
Qual a operação?  
+  
4 + 7 = 11
```

- b. No mesmo método, deverão fazer a validação dos valores que foram introduzidos. Tal validação pode ser realizada através de uma função que se encontra no ficheiro, chamado ValidarInput. Esta função deverá ser invocada a seguir às instruções que solicitam os números e o carácter. Ao realizar esta validação, são lançadas exceções referentes ao tipo de problema que ocorreu. Deve ser realizado o tratamento para cada uma destas exceções. Caso alguma das exceções ocorram, deve surgir uma mensagem descritiva do que ocorreu. **[15 pontos]**

```
Primeiro Número?  
1  
Segundo Número?  
0  
Qual a operação?  
/  
Ocorreu uma tentativa de divisão por 0
```

```
Primeiro Número?  
5  
Segundo Número?  
1  
Qual a operação?  
+  
O primeiro número é divisível por 5 ou o segundo número é divisível por 3
```

```
Primeiro Número?  
1  
Segundo Número?  
2  
Qual a operação?  
e  
O operador não é reconhecido
```

```
Primeiro Número?  
101  
Segundo Número?  
23  
Qual a operação?  
+  
Um dos valores introduzidos não se encontra entre 0 e 100
```

```
Primeiro Número?  
a  
O valor introduzido não é um número
```

Grupo IV – Extras (50 pontos)

1. Às 13h00, coloca a tua resolução num repositório do GitHub, enviando o link juntamente com a resolução. **[5 pontos]**
2. Documenta todas as funções com as quais interagiste durante o teste **[5 pontos]**
3. Pedra, Papel, Tesoura. Na solução encontrarás um projeto chamado GrupoIV. Nesse projeto existe uma classe chamada PedraPapelTesoura. Nela:
 - a. A função Start é o ponto principal para a execução do jogo. Nela deverás requisitar o número de jogadas necessárias para um jogador vencer (à melhor de...). Esta função deverá ser também responsável por, assim que um jogador vença, apresentar o resultado. **[5 pontos]**
 - b. Deverá ser solicitado o nome do jogador. **[2 pontos]**
 - c. Deverá ser validado se um jogador ou o computador venceu. **[10 pontos mediante eficiência]**
 - d. Deverá ser selecionada aleatoriamente a jogada do computador. Tal poderá ser feita recorrendo à função “Aleatório”, onde é gerado um número entre um mínimo e um máximo. **[5 pontos]**
 - e. Após solicitar a jogada do utilizador, e gerar a jogada do computador, deverá recorrer à função Display para apresentar a simulação da jogada em tempo real. Nesta função, são solicitados os argumentos p1 e p2, ou seja, a jogada de cada jogador (r, p, s). O terceiro parâmetro, *rounds* é apenas a quantidade de vezes que as mãos sobem e descem (o suspense). **[5 pontos]**
 - f. Após a jogada terminar é necessário identificar o vencedor. **[8 pontos]** Não esquecer que:
 - i. Pedra vence contra tesoura
 - ii. Tesoura vence contra papel
 - iii. Papel vence contra pedra
 - g. Após o fim da jogada, os números de jogadas vencidas por cada participante devem ser atualizados. **[5 pontos]**