

# Universidade do Algarve

## Programação Imperativa

### Ficha de exercícios online nº2

site mooshak: <http://deei-mooshak.ualg.pt/~pi>

Para sua informação, deixamos aqui o aviso que vigora durante as provas de avaliação, nomeadamente nas festas e nos exames:

#### Aviso Geral

- Use o computador **exclusivamente** para escrever, compilar e submeter programas C ao mooshak. **Tolera-se** a consulta dos acetatos das aulas teóricas, material das aulas práticas arquivadas no disco. Qualquer outro uso do computador que não seja resolver os problemas propostos, **não é autorizado** durante a festa e **qualifica-se como fraude**.
- Qualquer uso de material indevido (telemóveis, *chats*, pdfs etc...), qualquer comportamento indevido, não autorizado ou considerada fraude académica, etc...é sancionado com **reprovação imediata à UC de Programação Imperativa e será assinalada às autoridades académicas competentes**.
- A elegância e eficiência do código serão elementos de avaliação. Por exemplo, recorra à definição de funções sempre que justificado.
- A função **main** das diferentes soluções submetidas **deverá sempre devolver 0**. Qualquer outro valor será considerado anómalo pelo mooshak.
- **Deixe sempre uma linha em branco** no fim do ficheiro submetido.
- Um exercício pode exigir uma determinada solução (por exemplo, “*não usar o tipo float*”, “*não usar ciclos*”, etc.). Uma solução aceite pelo mooshak mas que não respeita estas exigências será avaliada **para metade da sua cotação**.
- Uma linha de input ou de output **termina sempre com \n**.
- Não há espaços no input ou no output a não ser os que estão explicitamente referidos no enunciado.
- **Nenhuma dúvida em tempo de prova será esclarecida.**

### Exercício 1 (A)

Imagine o caso das expressões aritméticas simples que envolvem sempre dois inteiros (do tipo `int`) e uma operação aritmética básica (uma das quatro operações básicas: `+`, `-`, `*`, `/`) Escreva um programa que lê uma linha (no `stdin`) com uma expressão aritmética deste tipo e que calcula e imprime o resultado inteiro no `stdout`. É garantido que nenhum valor introduzido ocasionará um `arithmetic overflow`.

**input:** uma linha com a expressão aritmética formatada da seguinte forma:  
um inteiro, um espaço, uma operação, um espaço, um inteiro.

Por exemplo:

10 + 123456

**output:** uma linha com o valor final quando é possível realizar a operação, ou a palavra `NO`, no caso contrário.

Por exemplo (em resposta ao exemplo de input anterior):

123466

□

## Exercício 2 (B)

Ajude o professor a calcular a nota de avaliação global de um aluno que passou por três testes.

Assim sendo, escreva um programa *C* que leia três notas inteiras de 0 a 20, *a*, *b* e *c*, dados pelo utilizador (via *stdin*) e que calcule avaliação final do aluno.

A primeira nota, a nota *a*, vale 50% da nota final. A nota *b* vale 30% da nota final e finalmente a nota *c* vale 20%.

A nota final é o arredondamento do valor obtido pelo cálculo da média ponderada.

É garantido que os 3 valores introduzidos no *input* sejam valores inteiros.

**input:** três linhas, com um inteiro em cada uma, representando uma nota introduzida.

Por exemplo:

10  
15  
18

**output:** uma linha com a nota final (um inteiro entre 0 e 20, ou então a palavra *NO* se uma qualquer anomalia no processo for detectada )

Por exemplo (em resposta ao exemplo de input anterior):

13

□

### Exercício 3 (C)

*Imagine um cronómetro electrónico ligado a um computador. Este cronómetro conta o tempo passado desde que está ligado em segundos.*

*Escreva um programa que lê este valor do `stdin` como um `long long` e diga quantos dias, horas, minutos e segundos (formato `DD:HH:MM:SS`) este valor representa.*

*É garantido que o valor introduzido no `input` seja um inteiro compatível com o tipo `long long`.*

**input:** *uma linha com um valor inteiro.*

*Por exemplo:*

1000000000

**output:** *uma linha com o tempo medido em dias, horas, minutos e segundos conforme o formato `DD:HH:MM:SS`, ou a palavra **NO** se uma qualquer anomalia for detectada.*

*Por exemplo (em resposta ao exemplo de input anterior):*

1157:9:46:40

*Este output significa que 1000000000 segundos são 1157 dias, 9 horas, 46 minutos e 40 segundos*

□

#### Exercício 4 (D)

Neste exercício queremos comparar dois números inteiros (tipo *int*) mas com um critério muito especial.

Primeiro, a noção de igualdade nesta comparação é a habitual (e.g. 12 é igual a 12).

Qualquer inteiro par é maior do que qualquer inteiro ímpar (e.g. 4 é maior do que 21).

Dentro dos pares, 0 compara-se com outros inteiros pares da forma natural. isto é, 0 é maior do que qualquer par negativo não nulo, e é menor do que qualquer par positivo não nulo.

Entre os pares não nulos, os que são divisíveis por 3 são maiores do que qualquer outro par não divisível por 3 (e.g. 18 é maior do que 20). Os pares divisíveis por 3 comparam-se entre si de forma crescente, ou seja pela ordem natural (e.g. 12 é menor do que 18). Os restantes pares comparam-se entre si de forma decrescente (e.g. 8 é menor do que 4).

Os ímpares divisíveis por 5 são maiores do que os outros ímpares (não divisível por 5). Assim, por exemplo 25 é maior do que 33. Entre si, os ímpares divisíveis por 5 comparam-se de forma de crescente (ou seja, a ordem natural). Assim 25 é menor do que 35. Nos restantes ímpares, a comparação é feita de forma decrescente. Por exemplo 11 é maior do que 33.

É garantido os dois valores introduzidos no *input* sejam valores inteiros compatíveis com o tipo *int*.

**input:** duas linhas com um valor inteiro em cada uma. Por exemplo:

```
14263
16
```

**output:** Uma linha com a palavra *MENOR* se o primeiro inteiro for menor do que o segundo. Ou então com a palavra *IGUAL* se os dois inteiros forem iguais. Ou então a palavra *MAIOR* no caso restante. Por exemplo (em resposta ao exemplo de input anterior):

```
MENOR
```

□

### Exercício 5 (E)

A tarefa neste exercício é ordenar 3 valores inteiros dados na entrada. Para saber se a ordenação pretendida é por ordem crescente ou por ordem decrescente, junta-se um inteiro (como primeiro valor fornecido).

Se este valor for positivo ou nulo, pretende-se a ordenação crescente. Se este primeiro inteiro for negativo, pretende-se a ordenação decrescente.

**Nota:** Espera-se que a ordenação seja feita **recorrendo a estruturas de decisão**, e não algoritmos clássicos que operam sobre vectores.

**input:** uma única linha com 4 inteiros separados entre eles por um único espaço. É garantido que estes valores numéricos são compatíveis com o tipo `int` do C.

Por exemplo:

-15 10 14 -5

**output:** uma única linha com os 3 últimos valores do input ordenados conforme o critério estabelecido pelo primeiro inteiro. Um único espaço separa os inteiros entre si.

Por exemplo (em resposta ao exemplo de input anterior):

14 10 -5

□

### Exercício 6 (F)

A seguradora *ERIS* oferece aos seus clientes um seguro da habitação contra todos os riscos cuja mensalidade é calculada tendo em conta, os anos totais desde que o cliente é cliente (os anos de fidelidade), os anos sem incidente desde o ultimo incidente que a seguradora cobriu, e a idade.

A mensalidade segue as regras seguintes:

- A cada 5 anos completos de fidelidade o cliente tem 1% de desconto;
- A cada 5 anos completos sem incidentes, o cliente tem 1% adicional de desconto;
- O cliente tem uma penalização de 5% por cada década que tem (arredondado sempre por cima, alguém com 43 anos conta como tendo 5 décadas, alguém com 23 conta como tendo 3 décadas).

**input:** uma linha com os seguintes valores separados entre eles por um espaço:

- a mensalidade bruta em euros, até ao cêntimo, calculada pela seguradora *Eris* com base na habitação que quer assegurar;
- um valor inteiro que representa os anos como cliente;
- um valor inteiro que representa os anos que tem desde o ultimo incidente;
- um valor inteiro representando a idade do cliente.

Por exemplo:

1100.50 12 4 43

**output:** uma linha com o valor final arredondado ao cêntimo de euro quando é possível realizar a operação, ou a palavra *NO*, no caso contrário.

Por exemplo (em resposta ao exemplo de input anterior):

1353.62

Porque só tem 2% de bonificação por ser cliente há 12 anos (duas vezes 5 anos completos), 0% de bonificação porque teve o ultimo incidente assegurado a 4 anos. e tem uma penalização de 25% por contar como tendo 5 décadas (43 arredondado a 50). No total são 23% a juntar ao valor mensal estimado:  $1100.50 + 23\%$ .

□

### Exercício 7 (G)

*Um dos paradoxos das sociedades modernas é o grande número de pessoas obesas. Ser obeso não é o mesmo que ser gordo. De facto, como diz o povo, “gordura é formosura”.*

*Por outro lado, obesidade é doença. Tecnicamente uma pessoa é obesa se o seu índice de massa corporal for maior ou igual a 32.4, sendo mulher, e maior ou igual a 31.2, sendo homem. O índice de massa corporal é o quociente do peso expresso em quilogramas pelo quadrado da altura expressa em metros.*

*Sendo assim, as unidades do índice de massa corporal são quilogramas por metro quadrado.*

**input:** *uma linha com uma letra ('M' para mulher, 'H' para homem), um valor de vírgula flutuante para indicar o peso em quilogramas, e um inteiro para representar a altura em centímetros. Estes valores são separados por um espaço.*

*É garantido que o primeiro valor introduzido é um valor de tipo `char`, que o peso é dado na forma de um valor compatível com o tipo `double` e que a altura é dada com um valor de tipo `int`.*

*Por exemplo:*

M 85.2 177

**output:** *uma única linha com dois elementos. O primeiro é o índice corporal, dado como `double` com **exatamente 3 casas decimais** e a palavra **YES** se for considerado obeso, ou a palavra **NO** se não for.*

*Se por uma razão qualquer o cálculo não pode ser realizado, o output é uma linha com a palavra **NO**.*

*Por exemplo (em resposta ao exemplo de input anterior):*

27.195 NO

□



## Exercício 8 (H)

*O bar da Faculdade resolveu implementar um programa de fidelização para refeições, para atrair mais estudantes. O valor original da refeição para estudante é 2.49 euros. A politica de fidelização funciona da seguinte forma:*

- *cada uma das primeiras 50 refeições custa o valor original, isto é 2.49 euros;*
- *a partir da 51.<sup>a</sup> refeição, há um desconto de 5% sobre o valor original, isto é, cada refeição custará 2,37 euros (isto é 5% mais barato do que o valor original (2.3655 euros, mas arredondado ao cêntimo mais próximo, logo 2.37));*
- *a partir da 101.<sup>a</sup> refeição há um desconto de 7% sobre o preço anterior exacto, isto é, 2.3655 - 7%. Ou seja, o preço é 2.199915 euros mas o estudante pagará 2.20 euros (por arredondamento ao cêntimo mais próximo);*
- *a partir da 201.<sup>a</sup>, há um novo desconto de 10% sobre o preço anterior, isto é o preço é 2.199915 -10%, Mais uma vez, o estudante pagará o valor arredondado ao cêntimo.*
- *a partir da 301.<sup>a</sup> refeição aplica-se um desconto suplementar, mas final, de 15%, aplicável a todas refeições acima das 301 (incluída).*

*Esta regra aplica-se até o estudante terminar o curso, independentemente do tempo que demorar. Queremos calcular quanto terá gasto o estudante em alimentação no bar da Faculdade, em função do número de refeições que fez.*

**input:** *uma única linha com um inteiro que representa o numero total de refeições que comeu na cantina durante o seu curso. É garantido que este valor é compatível com o tipo `int` do C. Por exemplo:*

102

**output:** *uma única linha a soma total que gastou na cantina, em euros (até ao cêntimo), ou a palavra `NO` se houver alguma anomalia detectada.*

*Por exemplo (em resposta ao exemplo de input anterior):*

247.40

□

### Exercício 9 (I)

Ajude o professor a calcular a nota de avaliação global de um aluno que passou por três testes.

Assim sendo, escreva um programa *C* que leia três notas inteiras de 0 a 20, *a*, *b* e *c*, dados pelo utilizador (via ***stdin***) e que calcule avaliação final do aluno.

As notas *a* e *b* são as notas dos dois testes da avaliação contínua. A nota *a* representa 40% da nota da avaliação contínua. A nota *b* representa os 60% restantes.

Chamemos *d* à nota inteira resultante da avaliação contínua arredondada à unidade mais próxima.

- Se *d* for maior ou igual a 10 em 20, o aluno em causa passou à unidade curricular. Neste caso a nota *c* do exame que teve de fazer é considerada uma nota de melhoria. Assim a nota final é a maior das duas notas (isto é ou *c* ou *d*).
- No caso contrário, o aluno precisa de ir a exame para obter aprovação, e assim a nota final é simplesmente a nota *c*.

É garantido que os 3 valores introduzidos no ***input*** sejam valores inteiros compatíveis com o tipo ***int***.

**input:** três linhas, com um inteiro em cada uma, representando uma nota introduzida.

Por exemplo:

10  
15  
12

**output:** uma linha com a nota final (um inteiro entre 0 e 20, ou então a palavra ***NO*** se uma qualquer anomalia no processo for detectada )

Por exemplo (em resposta ao exemplo de input anterior):

13

□

## Exercício 10 (J)

Numa defesa de estágio, o júri avalia o estagiário em 3 critérios que se debruçam sobre a qualidade do trabalho desenvolvido, do relatório e da apresentação. Designamos estes critérios de  $C_1$ ,  $C_2$  e  $C_3$ .

Em cada um dos 3 critérios, o júri dá uma de 5 notas: A, B, C, D e E. Depois desta avaliação, o júri propõe uma nota global de 0 a 20 que resulta da apreciação quantitativa global mas informal que o júri tem da qualidade do estágio realizada.

Mas as vezes, a proposta de nota global não está de acordo com as notas dadas nos critérios. Para ajudar a determinar a coerência da avaliação global com a avaliação detalhada por critério, o presidente de júri tem uma tabela:

Nota geral	Notas nos 3 critérios
20	a nota A nos três critérios
19	um B e dois A
de 17 a 18	dois B e um A, ou três B
de 14 a 16	pelo menos um C, com A ou B nos restantes
de 10 a 13	pelo menos um D, nenhum E
de 0 a 9	pelo menos um E

Tabela 1: Tabela de Avaliação

Assim, dar um 12 a um aluno que tem um dos critérios com nota E é **errado**.

Ajude o júri em validar a apreciação global tendo em conta a apreciação dada nos critérios  $C_1$ ,  $C_2$  e  $C_3$ .

**input:** três linhas com , em cada uma delas, uma das 5 letras 'A', 'B', 'C', 'D', ou 'E', representando uma nota introduzida para o critério  $C_1$  (resp.  $C_2$  e  $C_3$ ).

Por exemplo:

A  
C  
B  
17

**output:** uma linha

- com a palavra **NO** se uma qualquer anomalia no processo for detectada, ou
- com a palavra **OK** se a nota global é compatível como as notas dadas nos três critérios, ou, finalmente,
- com o texto **NOT OK** se a nota global não é compatível como as notas dadas nos três critérios.

Por exemplo (em resposta ao exemplo de input anterior):

NOT OK

□