Definições

PROGRAMAÇÃO E SISTEMAS DE INFORMAÇÃO – 1º ANO
CURSO PROFISSIONAL TÉCNICO DE GPSI

"Se os teus projetos forem para um ano, semeia o grão. Se forem para dez anos, planta uma árvore. Se forem para cem anos, educa o povo"

(Provérbio chinês)

Módulo 02

Programação Estruturada

Introdução

A decomposição de um problema é um aspeto muito importante na resolução de qualquer problema. Os subprogramas são uma ferramenta importante que possibilitam a divisão de um complexo algoritmo num determinado número de componentes, sendo cada componente implementado como um subprograma.

Dois tipos de subprogramas vão ser analisados:

- **Funções** (produz resultado)
- **Procedimentos** (não produz resultado)

As funções definidas pelo programador têm uma estrutura semelhante à dos procedimentos, apenas com a obrigação de devolver um determinado resultado.

Na linguagem Python, apenas são permitidas definições de funções.

Isto vai implicar que o modo como uma função é declarada e chamada num algoritmo é ligeiramente diferente do que se passa num procedimento.

Os parâmetros nos procedimentos podem ser de input (entrada) ou de output (saída), mas não são sempre de input ou output, exclusivamente. Muitas vezes servem para transferir informação para o subprograma, funcionando como parâmetro de input, mas recebem um novo valor do subprograma, que devolvem ao ponto de chamada, servindo então como parâmetro de output. Tais parâmetros são chamados parâmetros de input-output (entrada-saída).

A diferença entre uma função e um procedimento reflete-se também no modo como cada um destes dois tipos de subprogramas é chamado, nomeadamente:

- quando se trata de um procedimento, apenas tem que se escrever o respetivo identificador, com os eventuais argumentos que ele exija;
- quando se trata de uma função, torna-se necessário incluí-la numa instrução de escrita ou de atribuição.

Uma vantagem importantíssima dos subprogramas é que eles podem ser chamados quantas vezes quisermos dentro de um algoritmo. Podemos assim efetuar a mesma operação ou conjunto de operações em diferentes pontos de um algoritmo.

Mas, a vantagem dos subprogramas vai mais longe: podemos querer efetuar, em diferentes pontos de um algoritmo, o mesmo tipo de operações (o mesmo subprograma) mas com dados diferentes. Aí, surgem-nos novos elementos que os subprogramas podem utilizar - os parâmetros.

Os parâmetros são elementos semelhantes às variáveis, mas que são inseridos nos cabeçalhos dos subprogramas e que, depois, são usados nas chamadas a esses mesmos subprogramas. Os valores indicados no lugar dos parâmetros, quando se faz uma chamada a um subprograma, são chamados argumentos.

Um subprograma pode conter mais do que um parâmetro (inseridos dentro de parêntesis). Na altura em que é feita a chamada ao subprograma, é tida em consideração a ordem dos argumentos, bem como o tipo de dados a que pertencem e são necessários tantos argumentos quantos os parâmetros. Os argumentos utilizados nas chamadas a subprogramas podem ser não apenas valores diretos, mas também variáveis e expressões, desde que os valores sejam compatíveis com os correspondentes parâmetros.

Vantagens da utilização de subprogramas:

- limita-se a complexidade da solução global à custa da conjugação dos subprogramas;
- contribui-se para a clareza e legibilidade dos programas, pois o código que resolve cada tarefa está separado;
- identificam-se mais facilmente os dados necessários e os resultados produzidos por cada tarefa;
- evita-se a existência de código redundante (repetido), pois reunindo o código comum a tarefas diferentes pode-se depois reaproveitá-lo em pontos distintos do programa.

Proposta de atividades

1. Escreva uma função com o nome soma_quadrados que recebe um número inteiro positivo, n, e tem como valor a soma do quadrado de todos os números até n.

```
>>> soma_quadrados(3)
14
>>> soma_quadrados(5)
55
```

2. Elabore uma função que determine e apresente no monitor os n primeiros múltiplos de um número inteiro m (n e m devem ser parâmetros de entrada do função). Implemente o subprograma em linguagem Python, devendo construir um programa principal que se encarrega das tarefas de I/O e chamar a função criada.

3. Faça uma função que, sendo dado como parâmetro de entrada um número inteiro, calcule e apresente no monitor, todos os seus múltiplos inferiores a 100. Implemente o programa tendo em conta os requisitos (relativos à estrutura do programa) definidos em 1.

4. Elabore uma função que apresente no monitor todos os números pares entre dois números inteiros n e m (n<m). Os valores de n e m devem ser passados como parâmetros para o função e no caso de (m<n) deve ser apresentada a mensagem "Valores Inválidos!".

5. Implemente uma função que permita trocar o valor de duas variáveis, cujo valor foi especificado pelo utilizador. Teste-o num pequeno programa onde os valores sejam introduzidos no programa principal e a troca se processe na função.

6. Escreva uma função que, receba como parâmetros de entrada as medidas dos dois catetos de um triângulo retângulo, e devolva a medida da hipotenusa.

7. Se tivesse que devolver dois valores, a solução usada na questão anterior seria utilizável? Justifique. Mostre qual a alternativa, para que a função devolva a hipotenusa e o perímetro do triangulo.

8. Elabore um programa que determine o quadrado de um número inteiro n. O número n deve ser pedido ao utilizador e, através de uma função, devolver o seu quadrado.

9. Escreva um programa que permita a conversão de temperaturas em graus Centígrados para graus Fahrenheit e vice-versa. Para tal deverá escrever duas funções, uma para realizar a conversão para Celsius e outro para realizar a conversão para Fahrenheit. Cada um destes funções deverá receber como parâmetro o valor da temperatura a converter e mostrar no monitor a temperatura convertida (C = 5*(F – 32)/9; F = (9*C/5)+32).

10. Escreva um programa que utilize uma função chamada *multiplo*, que receba dois valores inteiros e devolva *true* ou *false* se um dos valores é múltiplo (ou não) do outro. O programa principal deverá aceitar dois números e dizer se os números introduzidos são ou não múltiplos um do outro.

11. Implemente uma função chamada fatorial que calcule n! (fatorial de n, em que n é um inteiro positivo). O valor de n deve ser um parâmetro da função e o valor calculado deve ser enviado para o exterior.

12. Adicione ao programa anterior uma função que chame a função fatorial para mostrar o fatorial dos números de 1 a 10.

BOM TRABALHO! TU ÉS CAPAZ! CONSTRÓI O TEU CONHECIMENTO…

Os professores da disciplina,
Andreia Quintal | Carlos Almeida