



Programação modular
Programação funcional

Programação modular



VIDA DE
PROGRAMADOR

real historia;
string sender = "Victor Lima de Andrade";

#2165





Programação modular

- A principal ideia da programação modular é que uma solução seja dividida em partes menores.
- Estas partes podem ser consideradas módulos.
- Módulos podem ser funções, classes, serviços, etc.
- A análise do problema e dos requisitos permite especificar os módulos e funções que serão implementadas.



Programação modular

- Questões importantes:
 - Quais funções ou classes serão criadas?
 - Como projetar funções ou classes reutilizáveis?
 - Como agrupar os componentes e torná-los acessíveis para diversos programas?
 - Como avaliar a qualidade dos componentes (acoplamento e coesão)?
 - Componentes permitem o desenvolvimento em equipes?



Programação modular

- As linguagens de programação possuem recursos para criação de funções e procedimentos.
- Funções sempre retornam algum valor ou objeto quando chamadas.
- Procedimentos são executados sem um retorno de valor.
- Quando uma função ou procedimento é chamado o controle passa para ele, retornando somente ao final da execução.



Programação modular

- Módulo geralmente é a denominação de um repositório (código fonte, biblioteca..) que armazena diversas funções ou procedimentos.
- Com a adoção do paradigma de programação orientada a objetos os módulos possuem também classes, objetos.
- As classes também implementam funções na forma de métodos.



Programação funcional

- A programação funcional pode ser considerada um paradigma ou estilo de programação.
- O ponto central é o uso de funções como elemento principal das soluções.
- Funções devem produzir resultados baseados somente nos dados que recebem.
- Funções não devem gerar alterações no estado global do sistema.



Programação funcional

- Os efeitos e resultados de uma função não podem depender do estado do sistema.
- Outro aspecto importante diz respeito a imutabilidade dos dados e variáveis, com funções que gerem novos resultados.
- Programação funcional enfatiza um estilo de programação declarativa.



Funções

- Funções são grupos de instruções que executam uma determinada tarefa e retornam um resultado.
- Antes da invocação de uma função ela deve ser definida.
- Uma função pode ser invocada diversas vezes em locais diferentes.
- Quando uma função é chamada, o fluxo de controle passa para ela até que finalize e retorne ao chamador.

Funções

```
def function_name():
```

```
...
```

```
...
```

```
# Start of program
```

```
...
```

```
...
```

```
function_name()
```

```
...
```

```
function_name()
```

```
...
```

```
def function_name(parameter list):  
    """docstring"""  
    statement  
    statement(s)
```



Funções

- Uma função é criada com uma instrução executável `def`, diferente de outras linguagens compiladas.
- Quando `def` é executado o interpretador cria um objeto e atribui a função a ele, desta forma, funções são objetos como outros elementos na linguagem.
- Instrução `def` é executada em tempo de execução.



Funções

- Instrução *return* devolve o objeto a quem chamou a função, entretanto, seu uso, porém, é opcional.
- Nomes seguem as mesmas convenções de variáveis.
- Parâmetros ou argumentos são opcionais.
- Cada função deve possuir um nome único, embora seja possível criar funções anônimas.
- Dois pontos(:) indicam o início das instruções que compõem o corpo da função.



Funções

- Não existe uma definição para o tipo de retorno de uma função, ao contrário de outras linguagens.
- Da mesma forma, os argumentos ou parâmetros também não são associados a um tipo específico.
- Neste contexto, qualquer função de Python é polimórfica.
- Um exemplo seria uma função que executa a seguinte expressão: `return x*y`.
- O que faz esta função? Qual seria o resultado?



Docstrings

- Com o aumento da complexidade das funções em um sistema, é importante documentar as mesmas.
- Docstrings são comentários dentro de funções que indicam o que ela faz, retorna e eventualmente o que são os argumentos recebidos.
- Uma docstring é delimitada por `""" ... """`.
- Para obter a documentação de uma função sem acessar o seu código-fonte é possível executar a instrução `print(nome_function.__doc__)`



Parâmetros e argumentos

- Uma função pode receber zero ou mais parâmetros.
- Parâmetro é a denominação usada para indicar a referência na definição da função: `func(param1, param2)`.
- Argumento é o termo para definir o valor passado para a função quando a mesma é chamada: `func(2, 3)`.
- Em muitos casos os termos são usados de maneira diferente ou um único termo para indicar ambos



Parâmetros e argumentos

- Ao definir uma função é possível definir quantos argumentos irá receber.
- Em alguns casos indica-se o nome e um valor default para o parâmetro.
- Caso o argumento não seja fornecido em um parâmetro é assumido o valor default.
- Caso não exista valor default, um erro é gerado pela falta de argumento.
- Se todos os parâmetros possuem valor default, é possível chamar a função sem argumentos.



Parâmetros e argumentos

- Se a associação dos valores de argumentos com parâmetros confia na ordem em que aparecem, a chamada deve enviar valores para todos.
- Quando os argumentos incluem o nome do parâmetro, a ordem de envio dos argumentos não é relevante.
- O uso de argumentos nomeados facilita a compreensão, melhorando a legibilidade do código.
- Ambientes de programação em geral mostram a lista dos parâmetros na escrita do código.

Desafio



- A instalação de painéis solares é uma alternativa para reduzir o valor da conta de luz. A partir desta instalação, o sistema produz uma determinada quantidade de Kw por dia de energia.
- Vamos desenvolver um sistema que permita acompanhar a produção diária e que permita consultas ao final de um mês ou dentro do mês.
- O sistema deve solicitar os valores da produção de cada um dos dias de um determinado mês.
- Deve solicitar ainda o valor consumido em cada dia, ou seja, quanto o usuário consumiu de energia naquele mesmo dia.

Desafio



- Os valores recebidos devem ser armazenados em uma estrutura (lista, dicionário, outro...).
- Após a leitura dos dados do mês, o usuário pode escolher uma das consultas abaixo:
 - Produção, consumo e diferença nos dias de um período (ex. do 01 ao dia 10).
 - Quantidade produzida e o dia em que ocorreu a maior produção.
 - Saldo final no mês, total produzido, total consumido e o saldo (produção – consumo).
- A solução deve ser desenvolvida com o uso de funções. Criar as funções e ao final fazer a chamada das funções com os devidos argumentos.