Curso: CST em Análise e Desenvolvimento de Sistemas Data: 05/06/2017
Disciplina: Programação I Professor: Vinicius Hartmann Ferreira

Modularização

Problemas podem ser resolvidos de diferentes formas. Uma maneira eficaz é dividir um determinado problema em diversos pequenos problemas. Além de permitir que cada etapa seja tratada independentemente também é possível reaproveitar as soluções. Considere um programa que, utilizando a biblioteca Turtle do Python, desenhe um quadrado (Quadro 1)

Quadro 1. Desenhando um quadrado com Turtle

```
#Importa a biblioteca Turtle
import turtle
#Inicializa a tartaruga
tarta = turtle.Pen()
#Define a cor de preenchimento
tarta.color('blue')
#Inicia preenchimento
tarta.begin_fill()
#Comandos
tarta.forward(100)
tarta.left(90)
tarta.forward(100)
tarta.left(90)
tarta.forward(100)
tarta.left(90)
tarta.forward(100)
tarta.left(90)
#Finaliza preenchimento
tarta.end_fill()
```

O resultado da execução deste trecho de código é a forma geométrica de um quadrado. Caso seja necessário replicar mais vezes um quadrado em um desenho será necessário repetir todo este código. Por meio da modularização se torna possível reaproveitar o código desenvolvido em diferentes momentos, conforme pode ser visto no Quadro 2.

Quadro 2. Função quadrado

```
import turtle
def quadrado():
    tarta = turtle.Pen()
    tarta.color('blue')
    tarta.begin_fill()
    tarta.forward(100)
    tarta.left(90)
    tarta.forward(100)
    tarta.left(90)
    tarta.forward(100)
    tarta.left(90)
    tarta.forward(100)
    tarta.left(90)
    tarta.end_fill()

quadrado()
quadrado()
```

A execução do código apresentado no Quadro 2 gera dois quadrados iguais, construídos um sobre o outro. Isto aconteceu porque o traçado é o mesmo nas duas situações. Porém, podemos modificar o traçado passando para uma mesma função parâmetros diferentes, conforme pode ser visto no Quadro 3.

Quadro 3. Função com parâmetros

```
import turtle

def quadrado(x):
    tarta = turtle.Pen()
    tarta.forward(x*100)
    tarta.left(90)
    tarta.forward(x*100)
    tarta.left(90)
    tarta.forward(x*100)
    tarta.left(90)
    tarta.forward(x*100)
    tarta.left(90)

quadrado(1)
quadrado(1.5)
```

As funções podem receber por parâmetro qualquer tipo de variável e também qualquer número de variáveis. Além disso, as funções também pode manipular os dados enviados e retornar os valores modificados, conforme pode ser visto no Quadro 4.

Quadro 4. Função com parâmetros e com retorno

```
def dobro (x):
    return x*2

valor = eval(input("Informe um valor:"))
print(dobro(valor))
```