

INTRODUÇÃO A LINGUAGEM DE PROGRAMAÇÃO



MINISTRANTE: ADILMAR COELHO DANTAS
MSC: CIÊNCIA DA COMPUTAÇÃO

INTRODUÇÃO

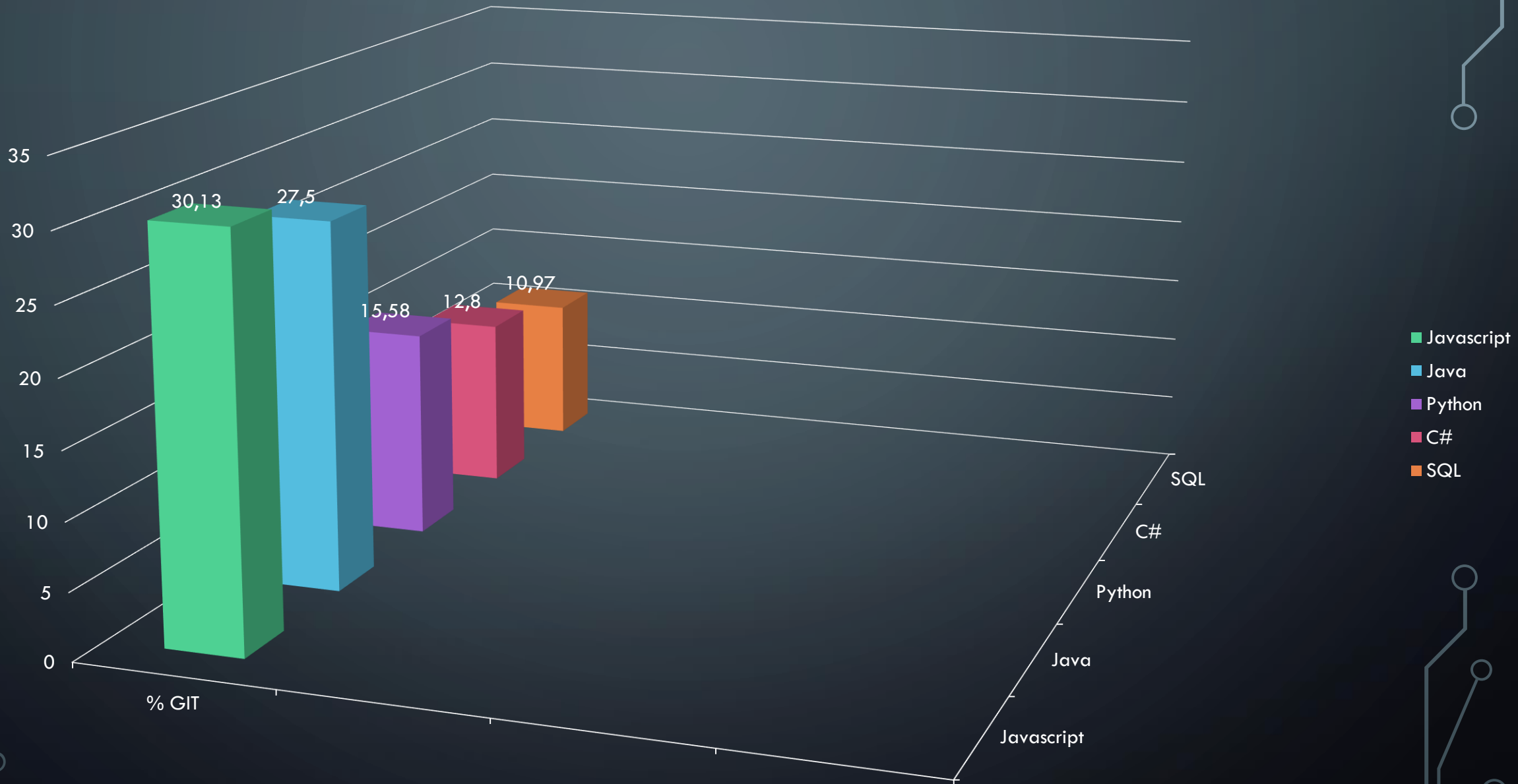
Python é uma linguagem de programação de alto nível interpretada, interativa, orientada a objetos e de alto nível. Foi criado por Guido van Rossum durante 1985-1990.

- Python é interpretado: o Python é processado no tempo de execução pelo intérprete. Você não precisa compilar seu programa antes de executá-lo. Isso é semelhante ao PERL e ao PHP.
- Python é interativo: você pode programar no prompt de Python e interagir com o intérprete diretamente para escrever seus programas.

INTRODUÇÃO

- Python é orientado a objetos: o Python suporta estilo orientado a objetos ou técnica de programação que encapsula código dentro de objetos.
- Python é um idioma para iniciantes: o Python é uma ótima linguagem para programadores de nível iniciante e oferece suporte ao desenvolvimento de uma ampla gama de aplicativos, desde o processamento de texto simples até navegadores WWW para jogos.

Linguagens de programação no cenário atual



PORQUE APRENDER PYTHON?

- Fácil de aprender: Python tem poucas palavras-chave, estrutura simples e uma sintaxe claramente definida. Isso permite que o aluno retire o idioma rapidamente.
- Fácil leitura: A exigência da formatação facilita a interpretação no momento da leitura do código fonte.
- Fácil de manter: Por exigir uma boa estrutura facilita na hora de realizar manutenções no fonte.
- Bibliotecas amplas: Possui diversas bibliotecas para as principais plataformas: Windows, MAC e UNIX.

PORQUE APRENDER PYTHON?

- Bancos de dados: o Python fornece interfaces para todos os principais bancos de dados comerciais.
- Programação GUI: o Python suporta aplicações GUI que podem ser criadas e portadas de maneira simples.
- Escalável: Permite trabalhar com aplicações de grande porte.

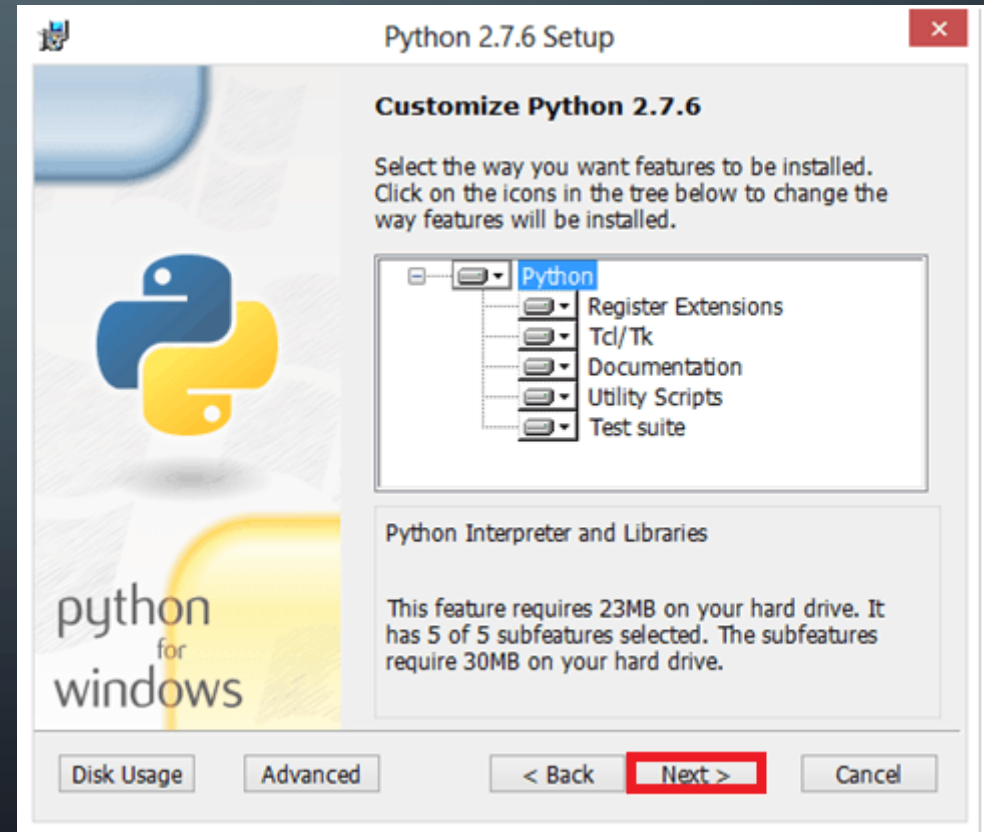
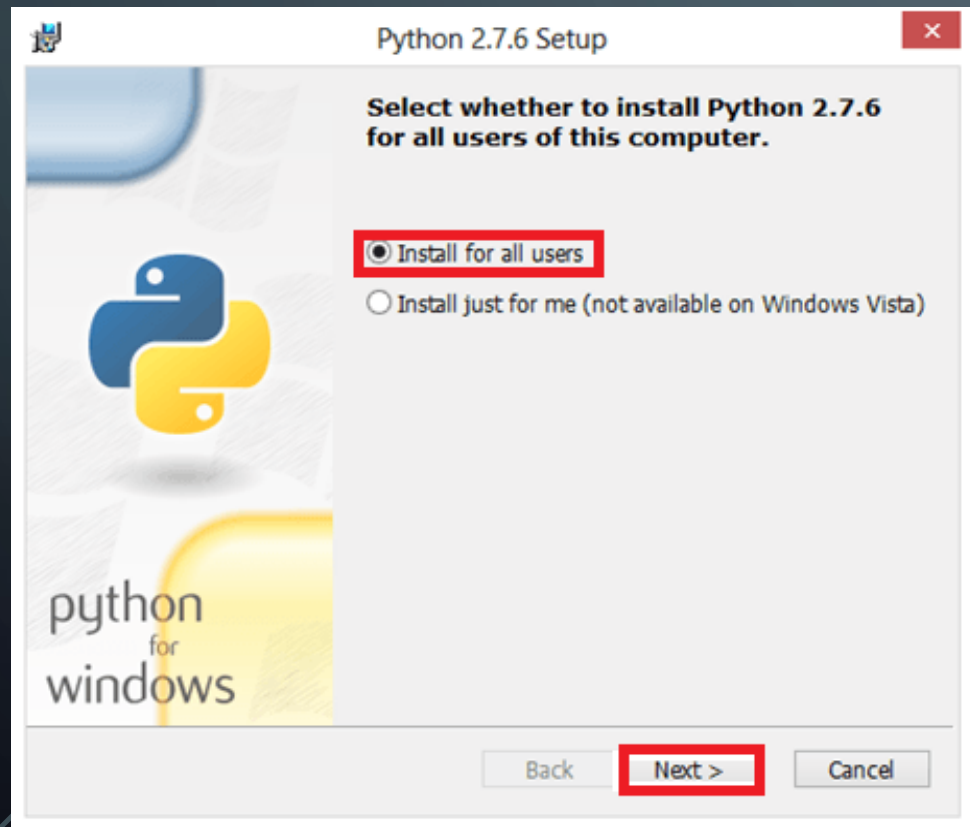


INSTALANDO O PYTHON

INSTALAÇÃO EM AMBIENTE WINDOWS

INSTALANDO O PYTHON 2.7

- Acesso: <https://www.python.org/downloads/release/python-2713/>



MEU PRIMEIRO PROGRAMA EM 3 PASSOS!



PROGRAMA 1 – “HELLO WORD “

PYTHON - EXERCÍCIO 1

Execute

main.py

```
1  #!/usr/bin/python
2
3  print "Hello, Python!"
```

☒ Result

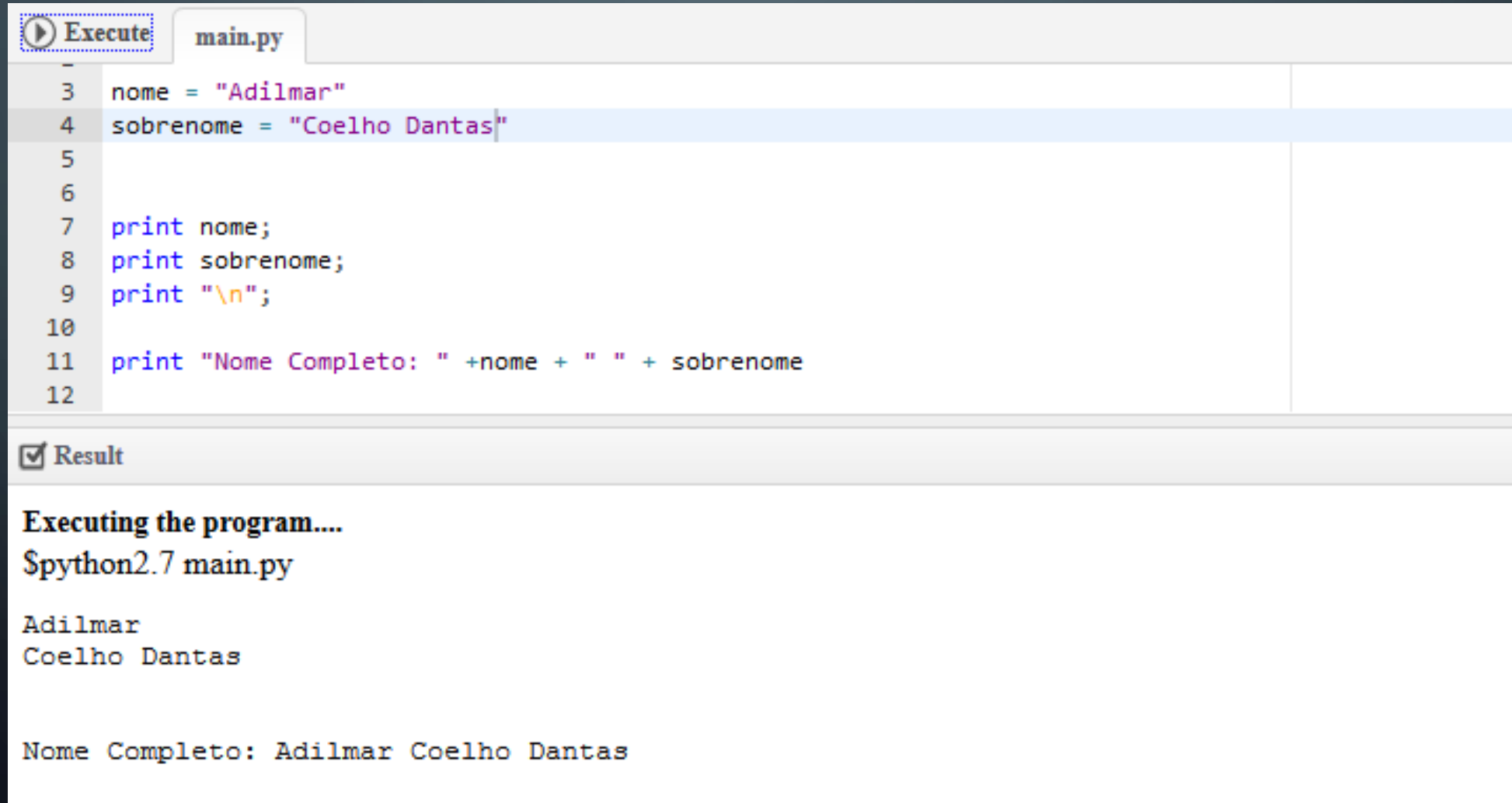
Executing the program....

\$python2.7 main.py

Hello, Python!

PROGRAMA 2 – VARIÁVEIS

PYTHON – EXER02.PY



The screenshot shows a Python IDE interface. At the top, there is a tab labeled 'main.py' and a button with a play icon and the text 'Execute'. Below the tab, the code is displayed with line numbers 3 through 12. Line 4 is highlighted. The code defines two variables, 'nome' and 'sobrenome', and prints them individually, followed by a blank line, and then concatenated. Below the code editor, there is a section titled 'Result' with a checked checkbox. It shows the command 'python2.7 main.py' and the output of the program: 'Adilmar', 'Coelho Dantas', and 'Nome Completo: Adilmar Coelho Dantas'.

```
3 nome = "Adilmar"
4 sobrenome = "Coelho Dantas"
5
6
7 print nome;
8 print sobrenome;
9 print "\n";
10
11 print "Nome Completo: " + nome + " " + sobrenome
12
```

☒ Result

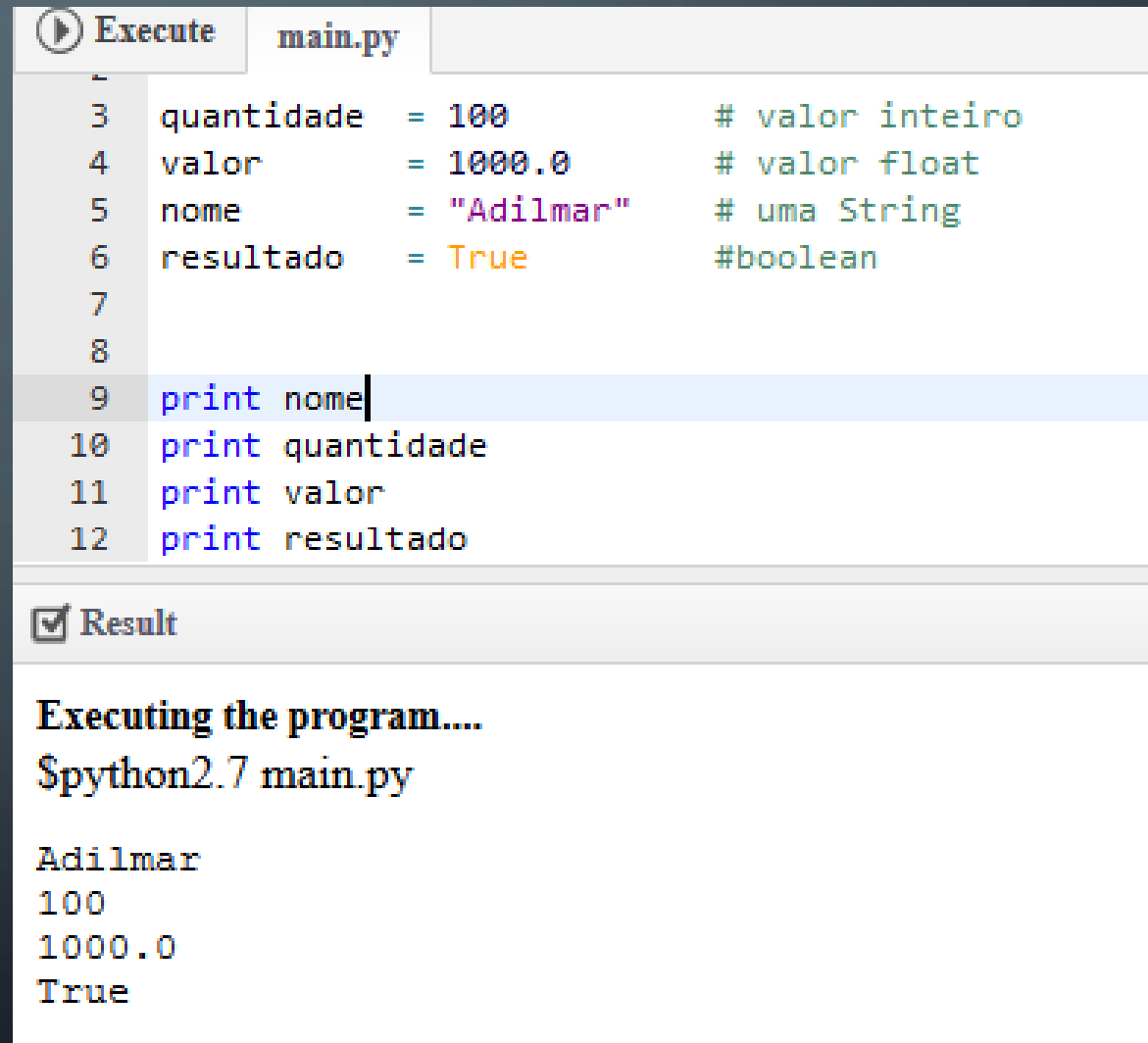
Executing the program....
\$python2.7 main.py

Adilmar
Coelho Dantas

Nome Completo: Adilmar Coelho Dantas

PROGRAMA 3 – TIPOS VARIÁVEIS

PYTHON – EXER03.PY



The screenshot shows a Python IDE window titled 'main.py'. The code defines four variables: 'quantidade' (integer), 'valor' (float), 'nome' (string), and 'resultado' (boolean). These variables are then printed out. The output window shows the results of the execution: 'Adilmar', '100', '1000.0', and 'True'.

```
1  
2  
3 quantidade = 100           # valor inteiro  
4 valor      = 1000.0        # valor float  
5 nome       = "Adilmar"     # uma String  
6 resultado  = True          #boolean  
7  
8  
9 print nome  
10 print quantidade  
11 print valor  
12 print resultado
```

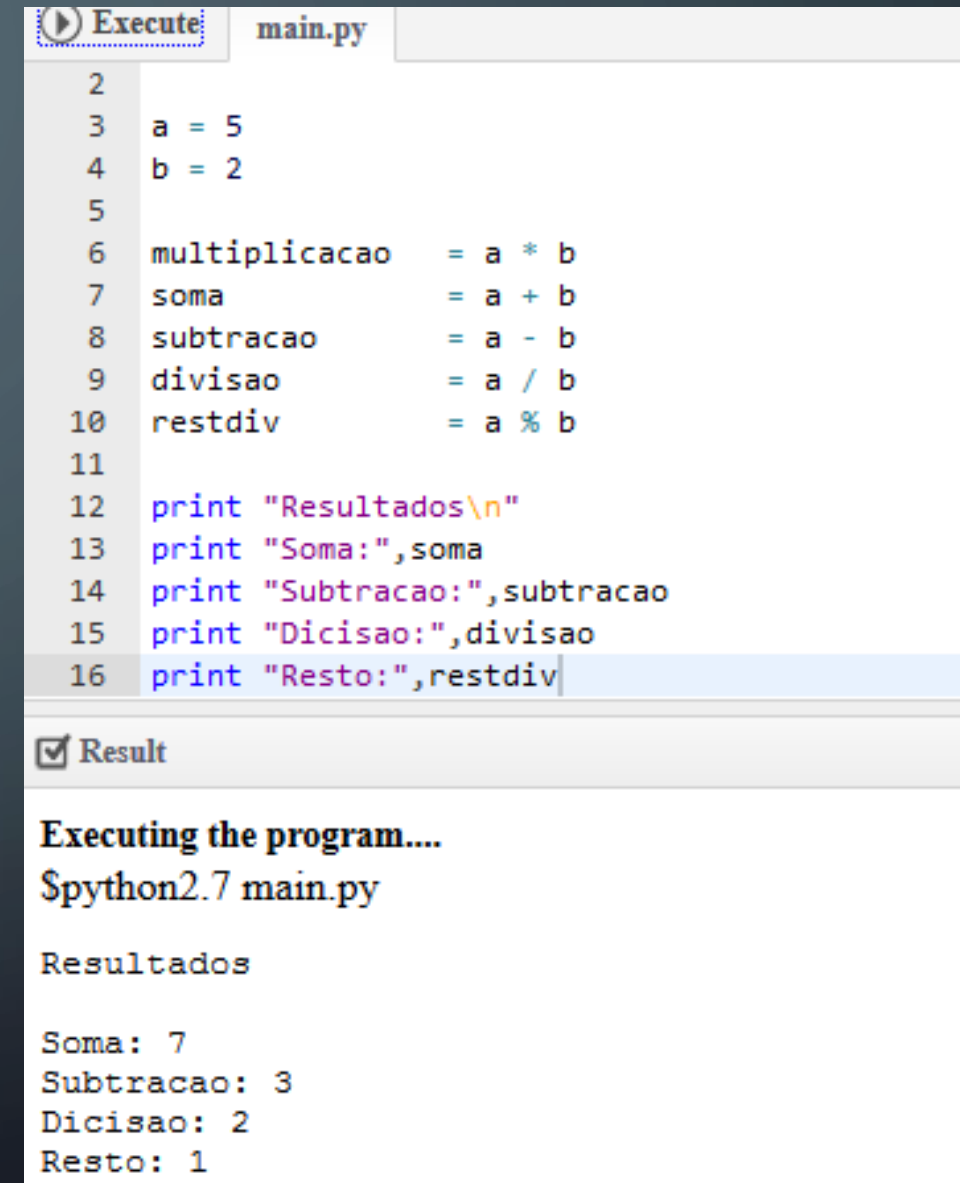
☒ Result

Executing the program....
\$python2.7 main.py

Adilmar
100
1000.0
True

PROGRAMA 4 – OPERADORES ARITMÉTICOS

PYTHON – EXER04.PY



```
2
3 a = 5
4 b = 2
5
6 multiplicacao = a * b
7 soma         = a + b
8 subtracao    = a - b
9 divisao      = a / b
10 restdiv      = a % b
11
12 print "Resultados\n"
13 print "Soma:",soma
14 print "Subtracao:",subtracao
15 print "Dicisao:",divisao
16 print "Resto:",restdiv
```

☒ Result

Executing the program....
\$python2.7 main.py

Resultados

Soma: 7
Subtracao: 3
Dicisao: 2
Resto: 1

OUTROS OPERADORES

== (operador de comparação)

E (and)

OU (or)

```
1 nome = input("Digite seu nome: ")
2 altura = input("Digite sua altura: ")
3
4 if(nome == "adilmar" or nome == "pedro"):
5     print("nome valido")
6
7 if(nome == "adilmar" and altura == "1.78"):
8     print("usuario cadastrado")
9 else:
10    print("usuario nao cadastrado")
```

(== , and, or)

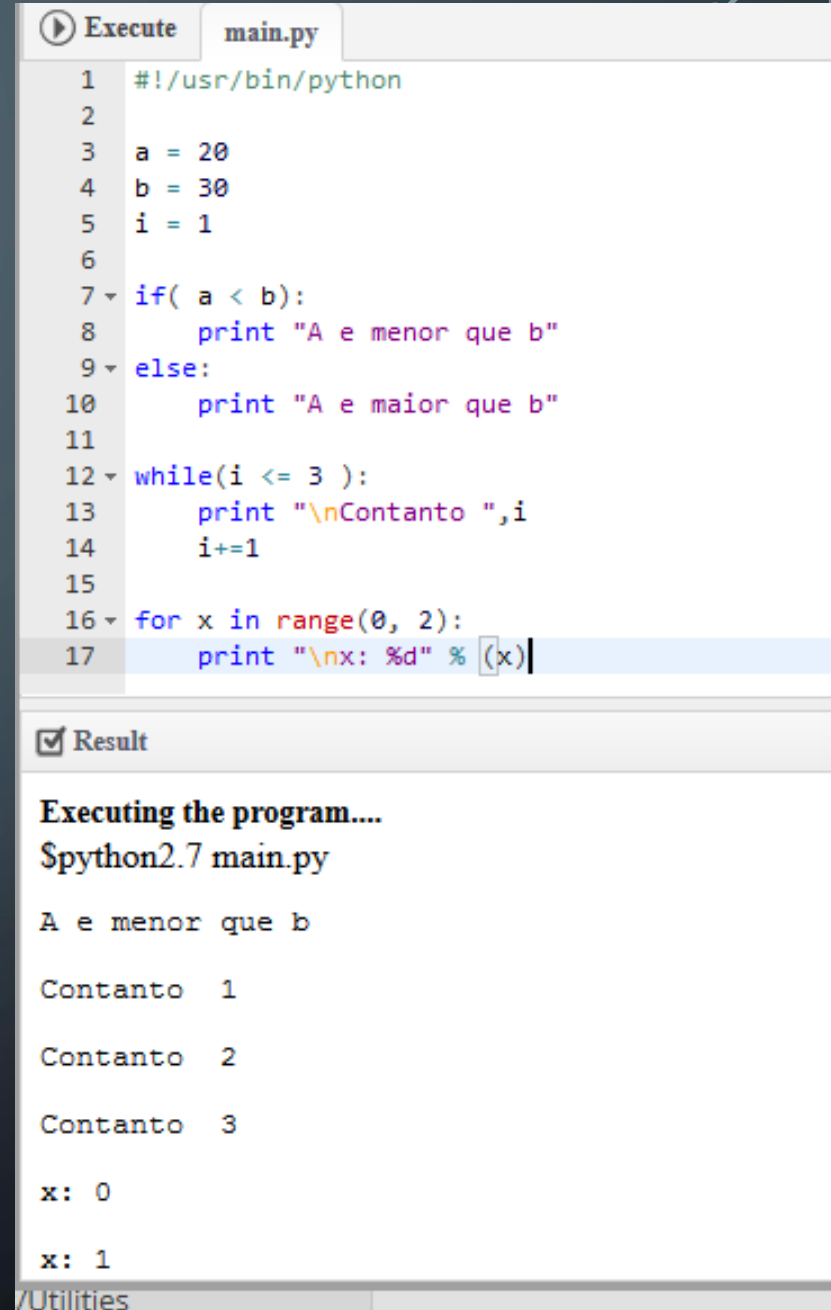
PROGRAMA 5 – ESTRUTURAS CONDICIONAIS

- Chamamos de estrutura condicional as instruções para testar se uma condição é verdadeira ou não.
- Estas estruturas condicionais podem ser associados com as estruturas que se repetem, após o cumprimento da condição, chamamos essas estruturas de repetição.

(if, while, for)

PROGRAMA 5 – OPERADORES ARITMÉTICOS E ESTRUTURAS DE REPETIÇÃO

PYTHON – EXER05.PY

A screenshot of a Python IDE window titled 'main.py'. The code editor shows a Python script with 17 lines. Line 1 is a shebang. Lines 3-5 assign values to variables a, b, and i. Lines 7-10 use an if-else statement to compare a and b. Lines 12-14 use a while loop to print 'Contanto' followed by the value of i, incrementing i by 1 each time. Lines 16-17 use a for loop to print 'x' followed by its value, iterating from 0 to 1. Below the code editor, there is a 'Result' section with a checked checkbox. It contains the text 'Executing the program....', 'Spython2.7 main.py', and the output of the program: 'A e menor que b', 'Contanto 1', 'Contanto 2', 'Contanto 3', 'x: 0', and 'x: 1'. The status bar at the bottom shows 'Utilities'.

PROGRAMA 6 – INPUT DE PARÂMETROS

PYTHON – EXER06.PY

input.py - C:\Users\Adilmar Dantas\Desktop\input.py

File Edit Format Run Options Windows Help

```
# -*- coding: cp1252 -*-
nome = raw_input("Digite seu nome: ")
print "\nOiii,",nome," vamos conversar..."
ano = raw_input("\nEm que ano você nasceu ? ")
print "\nEntão, você tem ",2017-int(ano), "anos hoje."
```

Python Shell

File Edit Shell Debug Options Windows Help

Python 2.7 (r27:82525, Jul 4 2010, 07:32)

Type "copyright", "credits" or "license()"

>>> ===== RESTART: Shell =====

>>>

Digite seu nome: adilmar

Oiii, adilmar vamos conversar...

Em que ano você nasceu ? 1992

Então, você tem 25 anos hoje.

>>> |

VAMOS PRATICAR

Python – exer07.py

Desenvolva uma calculadora com as operações básicas (+ , - , * , / , ²)

*Dica: use as estruturas ou a função input

PROGRAMA 8 – MANIPULANDO ARQUIVOS .TXT

- Já imaginou ter que comunicar um equipamento eletrônico e salvar suas saídas, por exemplo com Arduino.
- Por se tratar de uma tarefa relativamente simples podemos utilizar um arquivo “.txt” ou qualquer outra extensão para gravar estes dados, vejamos os passos.
- Definir o arquivo: `fp = open(dados.txt')`
- Criar uma constante para ler as linhas do file: `lines = fp.read().split("\n")`
- Fechar o arquivo: `fp.close()`



PROGRAMA 8 – LENDO ARQUIVOS

PYTHON – EXER08.PY

7% arquivos.py - C:\Users\Adilmar Dantas\Desktop\arquivos.py

File Edit Format Run Options Windows Help

```
fp = open('C://dados.txt') # abre o arquivo
lines = fp.read().split("\n") # cria uma lista
print(lines)
fp.close() # close file
```

```
print "\nFazendo de outra maneira\n"
```

```
with open('C://dados.txt') as fp:
    for line in fp:
        print line
```

7% Python Shell

File Edit Shell Debug Options Window

Python 2.7 (r27:82525, Jul 4 2008)
32

Type "copyright", "credits" or

>>> =====

>>>

['adilmar 25', 'pedro 33']

Fazendo de outra maneira

adilmar 25

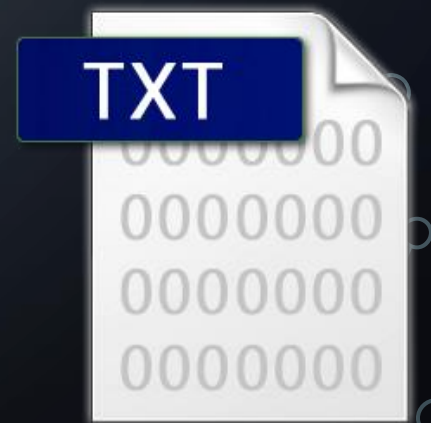
pedro 33

>>> |

PROGRAMA 9 – MANIPULANDO ARQUIVOS

.TXT - ESCRITA

- Assim como fizemos para ler o arquivo podemos criar um arquivo e escrever nele nossas informações, veja.
- Definir o arquivo: `file = open("testfile.txt","w")`
- Escrever no file: `file.write("Hello World")`



PROGRAMA 9 – ESCRIVENDO ARQUIVOS

PYTHON – EXER09.PY

7% exer9.py - C:/Users/Adilmar Dantas/Desktop/exer9.py

File Edit Format Run Options Windows Help

```
file = open("dados.txt", "w")

file.write("Nome: Adilmar\n")
file.write("Telefone: 99309-0377\n")
file.write("Aniversário: 02/02\n")
file.write("Profissão: Analista Sênior\n")
print("Salvo")

file.close()
```

7% Python Shell

File Edit Shell Debug Options Wind

```
Python 2.7 (r27:82525, Jul 4
32
Type "copyright", "credits" or
>>> =====
>>>
Salvo
>>> |
```

PROGRAMA 10 – CRIANDO UMA AGENDA

- Agora vamos criar uma agenda de contatos com os seguintes métodos (inserir e listar) e os parâmetros :

- Nome
- Celular
- Fixo
- Email
- Idade (deve ser calculado de maneira automática a partir do ano de nascimento)
- Delimitador (-----)
- Obs: criar um arquivo com o nome agenda.txt para gravar os dados



PROGRAMA 10 – AGENDA ELETRÔNICA

PYTHON – EXER10.PY

```
agenda.py - D:\Dropbox\Doutorado\POSMEC\codes\agenda.py
File Edit Format Run Options Windows Help

op='0'

print ("::~::~:Agenda Eletronica::~:~::~:")
print ("\nLISTA DE OPÇÕES\n1-Inserir\n2-Listar")

while (op=='0'):
    opcao = raw_input("\nDigite uma opção: ")

    if (opcao=='1'):
        file = open("agenda.txt","a")
        op='1'
        print("\nOPÇÃO SELECIONADA: INSERIR CONTATO")
        nome = raw_input("\nInforme o nome: ")
        celular = raw_input("Informe o celular: ")
        fixo = raw_input("Informe o Fixo: ")
        email = raw_input("Informe o email: ")
        ano = raw_input("Informe o ano de nascimento: ")
        idade= 2017 - int(ano);

        file.write("Nome: "      +nome      +"\n")
        file.write("Celular: "   +celular   +"\n")
        file.write("Fixo: "      +fixo      +"\n")
        file.write("email: "     +email     +"\n")
        file.write("Idade: "     +str(idade) +"\n")
        file.write("-----\n")
        file.close()
        op='0'

    if (opcao=='2'):
        file = open("agenda.txt","a")
        op='1'
        print("\nOPÇÃO SELECIONADA: LISTAR CONTATOS")
        with open("agenda.txt") as agenda:
            for contato in agenda:
                print contato
        file.close()
```

Python Shell

File Edit Shell Debug Options Windows Help

Python 2.7 (r27:82525, Jul 4 2010, 07:43:08) [MSC v.1500 64 32

Type "copyright", "credits" or "license()" for more informat

>>> ===== RESTART =====

>>>

:~::~:Agenda Eletronica::~:~::~:

LISTA DE OPÇÕES

1-Inserir

2-Listar

Digite uma opção: 1

OPÇÃO SELECIONADA: INSERIR CONTATO

Informe o nome: José

Informe o celular: 9999999

Informe o Fixo: 333333

Informe o email: jose@bol.com.br

Informe o ano de nascimento: 1968

Digite uma opção: 2

OPÇÃO SELECIONADA: LISTAR CONTATOS

Nome: adilmair

Celular: 1

Fixo: 2

email: 3

Idade: 25

Nome: pedro

Celular: 2

PROGRAMA 11 – LISTAS, TUPLAS E DICIONÁRIOS

- Python – exer11.py

```
1  #!/usr/bin/python
2
3  lista = [ 'adilmar', 786 , 2.23, 'pedro', 70.2 ]
4
5  print("LISTA\n")
6
7  print lista          # Prints complete list
8  print lista[0]       # Prints first element of the list
9  print lista[1:3]     # Prints elements starting from 2nd till 3rd
10 print lista[2:]      # Prints elements starting from 3rd element
11
12 print("\nTUPLA\n")
13
14 tupla = ( 'adilmar', 786 , 2.23, 'pedro', 70.2 )
15 tinytuple = (123, 'jose')
16
17 print tupla          # Prints complete list
18 print tupla[0]       # Prints first element of the list
19 print tupla[1:3]     # Prints elements starting from 2nd till 3rd
20 print tupla[2:]      # Prints elements starting from 3rd element
21 print tinytuple * 2  # Prints list two times
22 print tupla + tinytuple # Prints concatenated lists
23 print("\nDicionario\n")
24
25 tinydict = {'nome': 'adilmar', 'codigo': 6734, 'departamento': 'TI'}
26 print tinydict       # Prints complete dictionary
27 print tinydict.keys() # Prints all the keys
28 print tinydict.values() # Prints all the values
```

LISTA

```
['adilmar', 786, 2.23, 'pedro', 70.2]
adilmar
[786, 2.23]
[2.23, 'pedro', 70.2]
```

TUPLA

```
('adilmar', 786, 2.23, 'pedro', 70.2)
adilmar
(786, 2.23)
(2.23, 'pedro', 70.2)
(123, 'jose', 123, 'jose')
('adilmar', 786, 2.23, 'pedro', 70.2, 123, 'jose')
```

Dicionario

```
{'codigo': 6734, 'departamento': 'TI', 'nome': 'adilmar'}
['codigo', 'departamento', 'nome']
[6734, 'TI', 'adilmar']
```

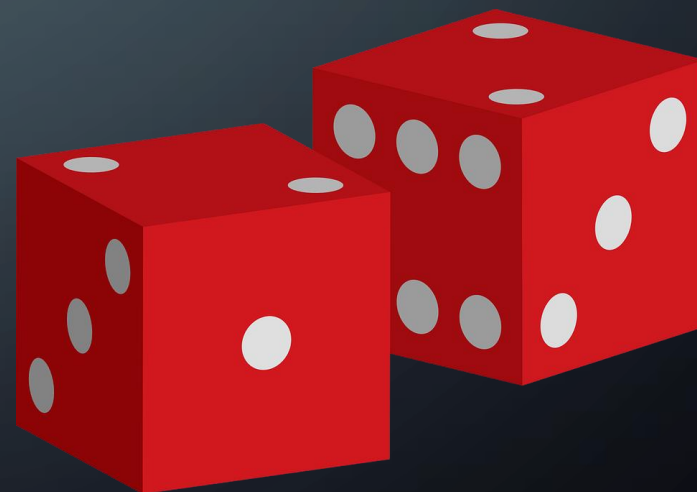
PROGRAMA 12 – FUNÇÕES

```
1  # Function definition is here
2
3  def funcao1():                #basic function
4      print "Essa e uma funcao" #print basic text
5
6  def soma(x,y):                #function sun (x,y) parameters
7      soma = x+y;               #calculate
8      print "A soma e ",soma    #return sun x+y
9
10 def verifica(x):              #function pair / odd
11     resto = x%2;               #calculate mod division by 2
12     if(resto == 0):
13         print "Numero par"
14     else:
15         print "Numero impar"
16
17 funcao1()    # call the function
18 soma(2,5)
19 verifica(5)
20
21 |
```

- Python – exer12.py

DESAFIO, JOGO SIMPLES !

- Agora que você já conhece a maioria dos conceitos de programação em Python que tal desenvolvermos um jogo de adivinhar o número sorteado pelo computador.
- Para isso vamos conhecer uma função reservada para fazer este sorteio, com as seguintes instruções.
- Precisa ser um número inteiro
- Compreendido de 1 a 100
- Para isso : `sorteado = randint(1, 100)`



CONTATOS - SOCIAL



Web: www.adilmar.com.br

E-mail: akanehar@gmail.com

Lattes: <http://lattes.cnpq.br/2462384793631673>

FB/IG @adilmarcoelho

