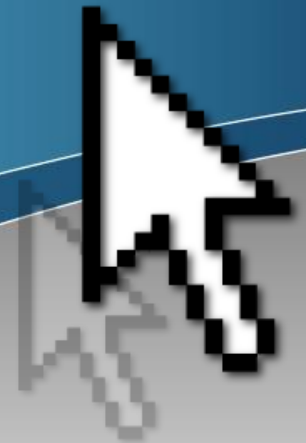


Programação de Computadores II

Revisão com codificação em C

Parte 2 de 2



Prof. Francisco Borges

Agenda

- Vetores
- Matrizes
- Cadeias de Caracteres
- Modularização
- Funções Pré-definidas
- Funções Definidas pelo Programador
 - Funções que Retornam Valor
 - Funções que Não Retornam Valor
 - Passagem de Parâmetros

Vetores

- Também conhecido como *arranjos*
- Itens de dados relacionados do mesmo tipo, ou seja, coleção de dados do mesmo tipo
- Armazenado em posições consecutivas de memória

Vetores

- Usado quando existe a necessidade de armazenar vários dados do mesmo tipo e para o mesmo fim
- Evita a declaração de múltiplas variáveis
- Sintaxe para declaração:

<tipo> <nomeDoVetor> [<tamanho>];

- Exemplo:

int idadeFunc [10];

Vetores Unidimensionais

■ Exemplo 1

```
#include <stdio.h>

int main()
{
    const int TOTFUNC=3;
    float salario[TOTFUNC];
    printf("Informe o salario de %i funcionarios:\n",
           TOTFUNC);
    scanf("%f %f %f", &salario[0], &salario[1], &salario[2]);
    printf("\nSalarios informados:\n\t%.2f\n\t%.2f\n\t%.2f",
           salario[0], salario[1], salario[2]);
    return 0;
}
```

Vetores Unidimensionais

■ Exemplo

```
#include <stdio.h>
```

```
int main()
```

```
{
```

```
    const int TOTFUNC=3;
```

```
    float salario[TOTFUNC];
```

```
    printf("Informe o salario de %i funcionarios:\n",
```

```
    TOTFUNC);
```

```
    scanf("%f %f %f", &salario[0], &salario[1], &salario[2]);
```

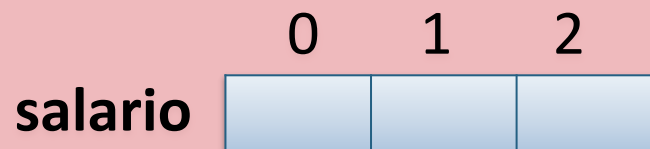
```
    printf("\nSalarios Informados:\n\t%.2f\n\t%.2f\n\t%.2f",
```

```
    salario[0], salario[1], salario[2]);
```

```
    return 0;
```

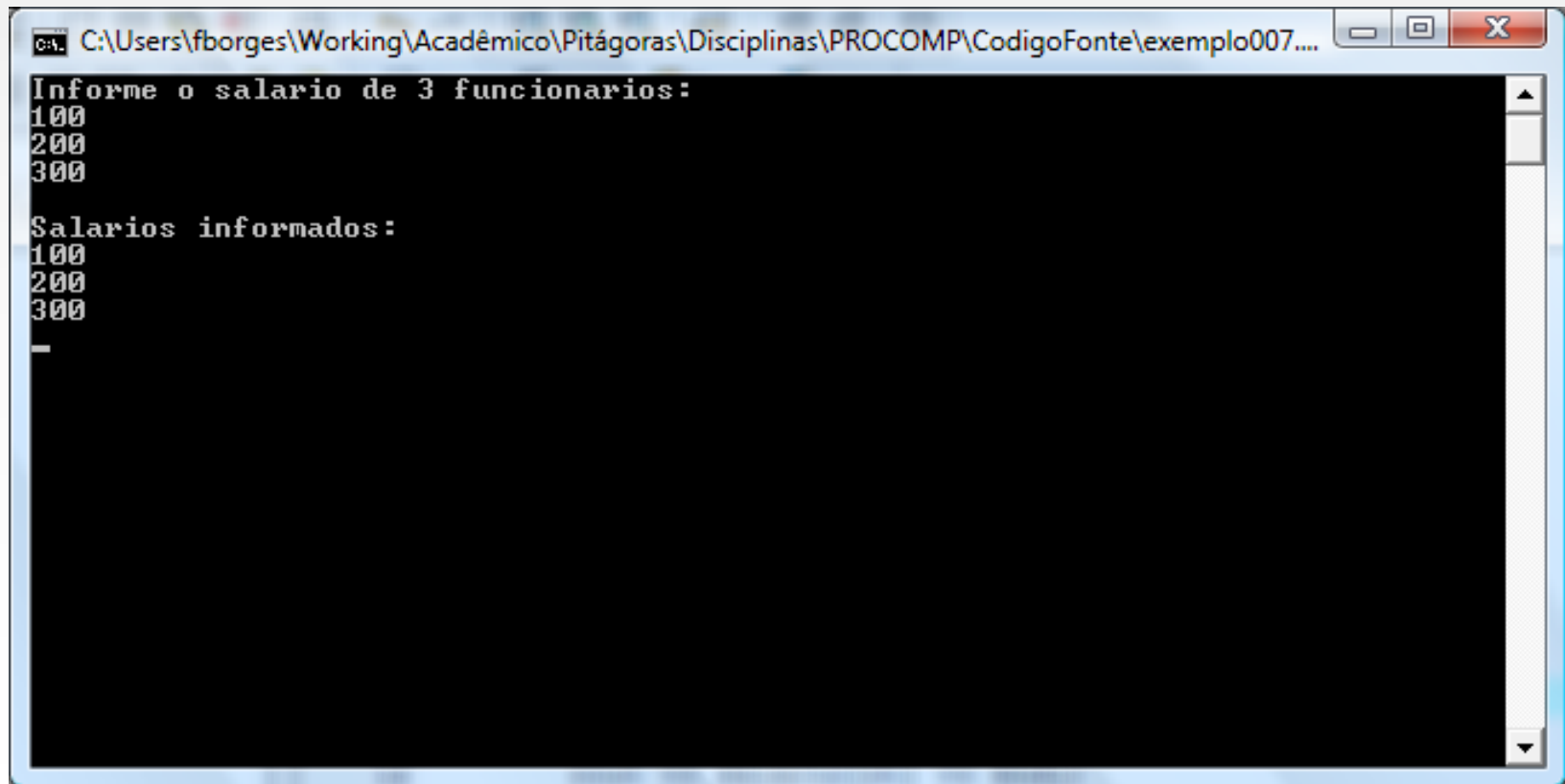
```
}
```

- Declaração de um vetor, chamado **salario** do tipo *float*, com três posições, ou seja, três elementos



Vetores Unidimensionais

- Resultado da execução do programa



```
C:\Users\fborges\Working\Acadêmico\Pitágoras\Disciplinas\PROCOMP\CodigoFonte\exemplo007....  
Informe o salario de 3 funcionarios:  
100  
200  
300  
  
Salarios informados:  
100  
200  
300  
-
```

Vetores Unidimensionais

■ Exemplo 3

- Uma empresa deseja dar um aumento de 15% aos seus dez funcionários. Sabendo que os salários dos funcionários estão armazenados em um vetor unidimensional, escreva um programa em C que:
 - Aplique o aumento desejado aos salários dos funcionários; e,
 - Mostre na tela o salário de todos os funcionários.

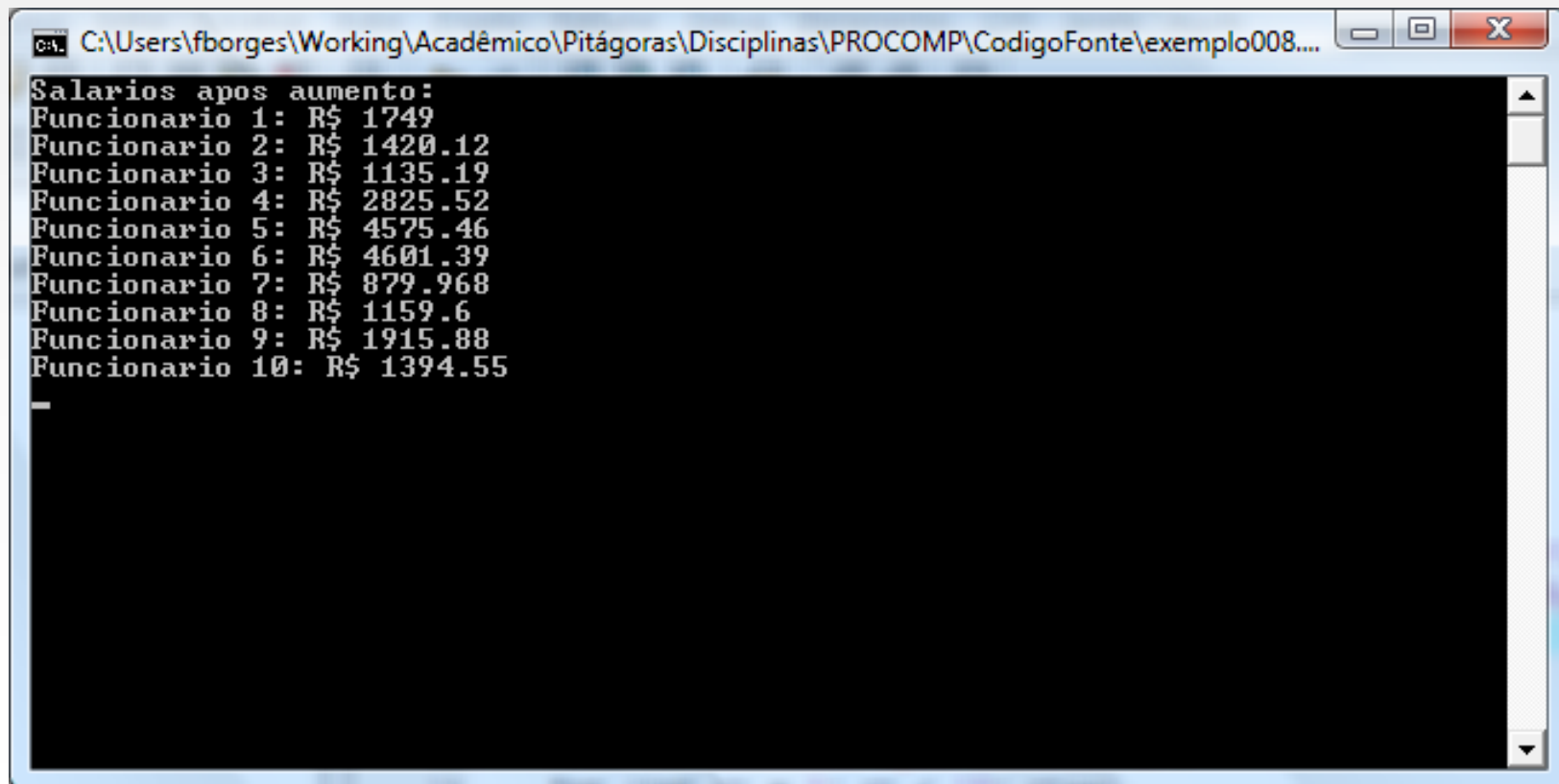
Vetores Unidimensionais

```
#include <stdio.h>

int main()
{
    double salario[10] = {1520.87, 1234.89, 987.12, 2456.97,
                          3978.66, 4001.21, 765.19, 1008.35,
                          1665.98, 1212.65};
    for (int i = 0; i < 10; i++)
        salario[i] = salario[i] * 1.15;
    printf("Salarios apos aumento:\n");
    for (int ct = 0; ct < 10; ct++)
        printf("Funcionario %i: R$ %.2f\n", ct + 1,
              salario[ct]);
    return 0;
}
```

Vetores Unidimensionais

- Resultado da execução do programa



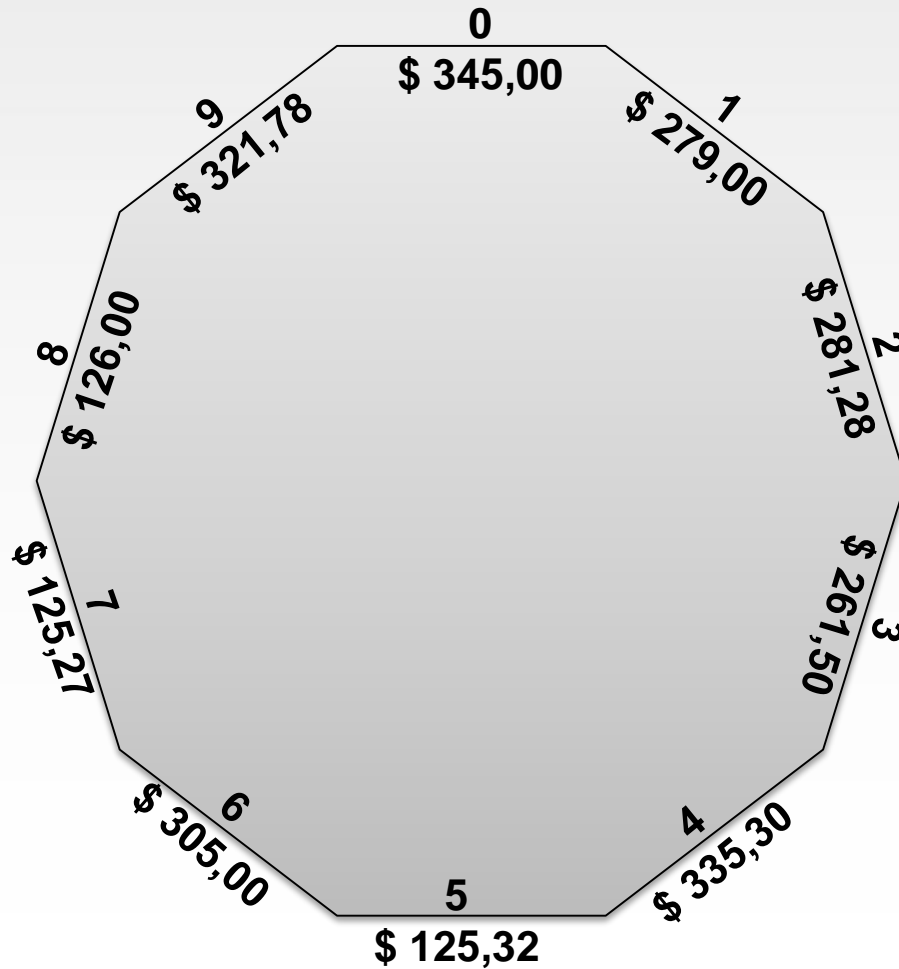
```
C:\Users\fborges\Working\Acadêmico\Pitágoras\Disciplinas\PROCOMP\CodigoFonte\exemplo008....  
Salarios apos aumento:  
Funcionario 1: R$ 1749  
Funcionario 2: R$ 1420.12  
Funcionario 3: R$ 1135.19  
Funcionario 4: R$ 2825.52  
Funcionario 5: R$ 4575.46  
Funcionario 6: R$ 4601.39  
Funcionario 7: R$ 879.968  
Funcionario 8: R$ 1159.6  
Funcionario 9: R$ 1915.88  
Funcionario 10: R$ 1394.55  
-
```

Vetores Unidimensionais

■ Exemplo 4

- Uma empresa acabou de adquirir um terreno de formato decagonal todo murado e deseja demolir o muro de um dos lados do terreno. A demolição possui um custo diferente para cada um dos muros. Sabendo que o custo da demolição de cada muro está armazenado num vetor unidimensional, mostre na tela o identificador do muro de menor custo.

Vetores Unidimensionais



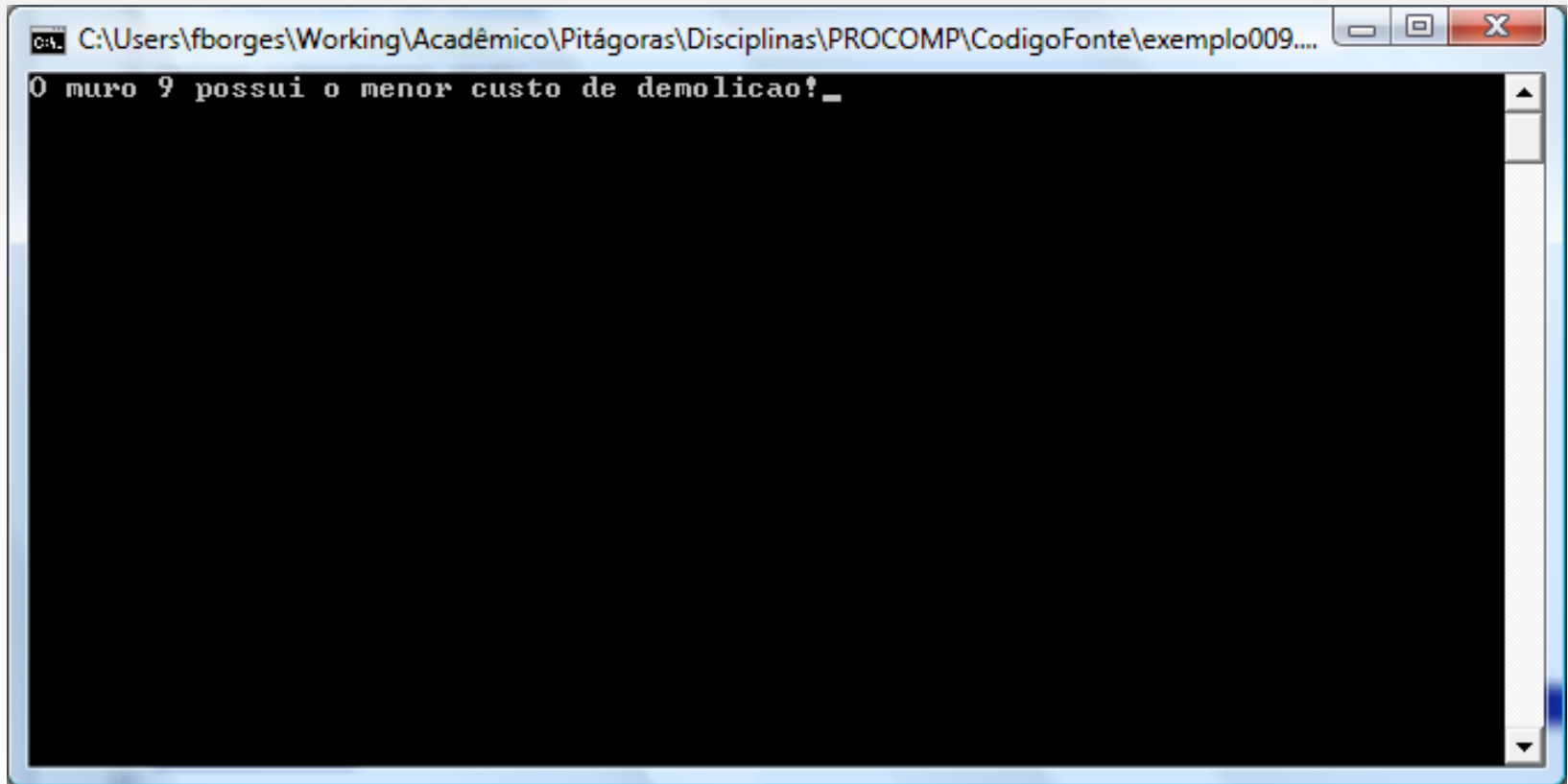
Vetores Unidimensionais

```
#include <stdio.h>
int main()
{
    double custoDem[10] = {345.00, 279.00, 281.28, 261.50,
                           335.30, 125.32, 305.00, 125.27,
                           126.00, 321.78};

    double menor = custoDem[0];
    int idMuro = 0;
    for (int i = 1; i < 10; i++){
        if (custoDem[i] < menor){
            idMuro = i;
            menor = custoDem[i];
        }
    }
    printf("O muro %d possui o menor custo!", idMuro);
    return 0;
}
```

Vetores Unidimensionais

- Resultado da execução do programa



A screenshot of a Windows command prompt window. The title bar shows the file path: C:\Users\fborges\Working\Acadêmico\Pitágoras\Disciplinas\PROCOMP\CodigoFonte\exemplo009.... The command prompt displays the output of a program: "O muro 9 possui o menor custo de demolicao!_" followed by a cursor. The window has a standard Windows interface with minimize, maximize, and close buttons in the title bar.

```
C:\Users\fborges\Working\Acadêmico\Pitágoras\Disciplinas\PROCOMP\CodigoFonte\exemplo009....  
O muro 9 possui o menor custo de demolicao!_
```

Vetores Multidimensionais

- Utilizam mais de um índice no seu endereçamento
- Matrizes podem ser representadas por meio de vetores multidimensionais, nesse caso, com duas dimensões
- Vetores com duas dimensões possuem dois índices e também são conhecidos como vetores bidimensionais
- Utilizados quando há a necessidade de algum tipo de organização que pressuponha a existência de mais de um índice

Vetores Multidimensionais

- Matriz (vetor bidimensional)

int a[3][4];

	Coluna 0	Coluna 1	Coluna 2	Coluna 3
Linha 0	a [0] [0]	a [0] [1]	a [0] [2]	a [0] [3]
Linha 1	a [1] [0]	a [1] [1]	a [1] [2]	a [1] [3]
Linha 2	a [2] [0]	a [2] [1]	a [2] [2]	a [2] [3]

Vetores Multidimensionais

■ Exemplo 1

- Escreva um programa em C que preencha um vetor bidimensional 3 x 3 com inteiros digitados pelo usuário.

Vetores Multidimensionais

```
#include <stdio.h>

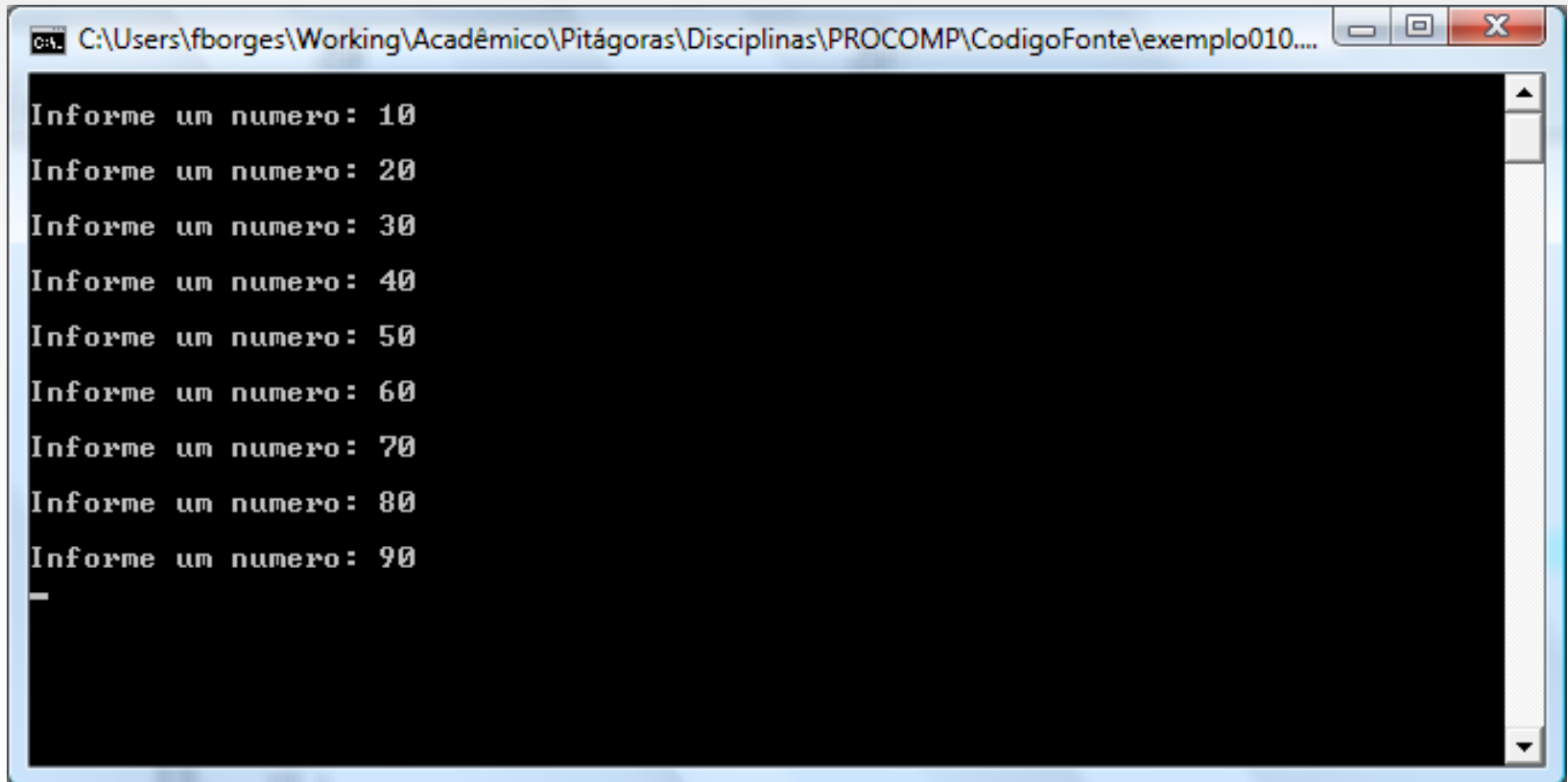
int main() {
    int numeros[3][3];

    for (int ctFor = 0; ctFor < 3; ctFor++) {
        for (int ctMat = 0; ctMat < 3; ctMat++) {
            printf("\nInforme um numero: ");
            scanf("%d", &numeros[ctFor][ctMat]);
        }
    }

    return 0;
}
```

Vetores Multidimensionais

- Resultado da execução do programa



```
C:\Users\fborges\Working\Acadêmico\Pitágoras\Disciplinas\PROCOMP\CodigoFonte\exemplo010....  
Informe um numero: 10  
Informe um numero: 20  
Informe um numero: 30  
Informe um numero: 40  
Informe um numero: 50  
Informe um numero: 60  
Informe um numero: 70  
Informe um numero: 80  
Informe um numero: 90  
_
```

Vetores Multidimensionais

■ Exemplo 2

- Escreva um programa em C que inicialize uma matriz 2 x 4 com números inteiros e calcule a média entre todos os números.

Vetores Multidimensionais

```
#include <stdio.h>

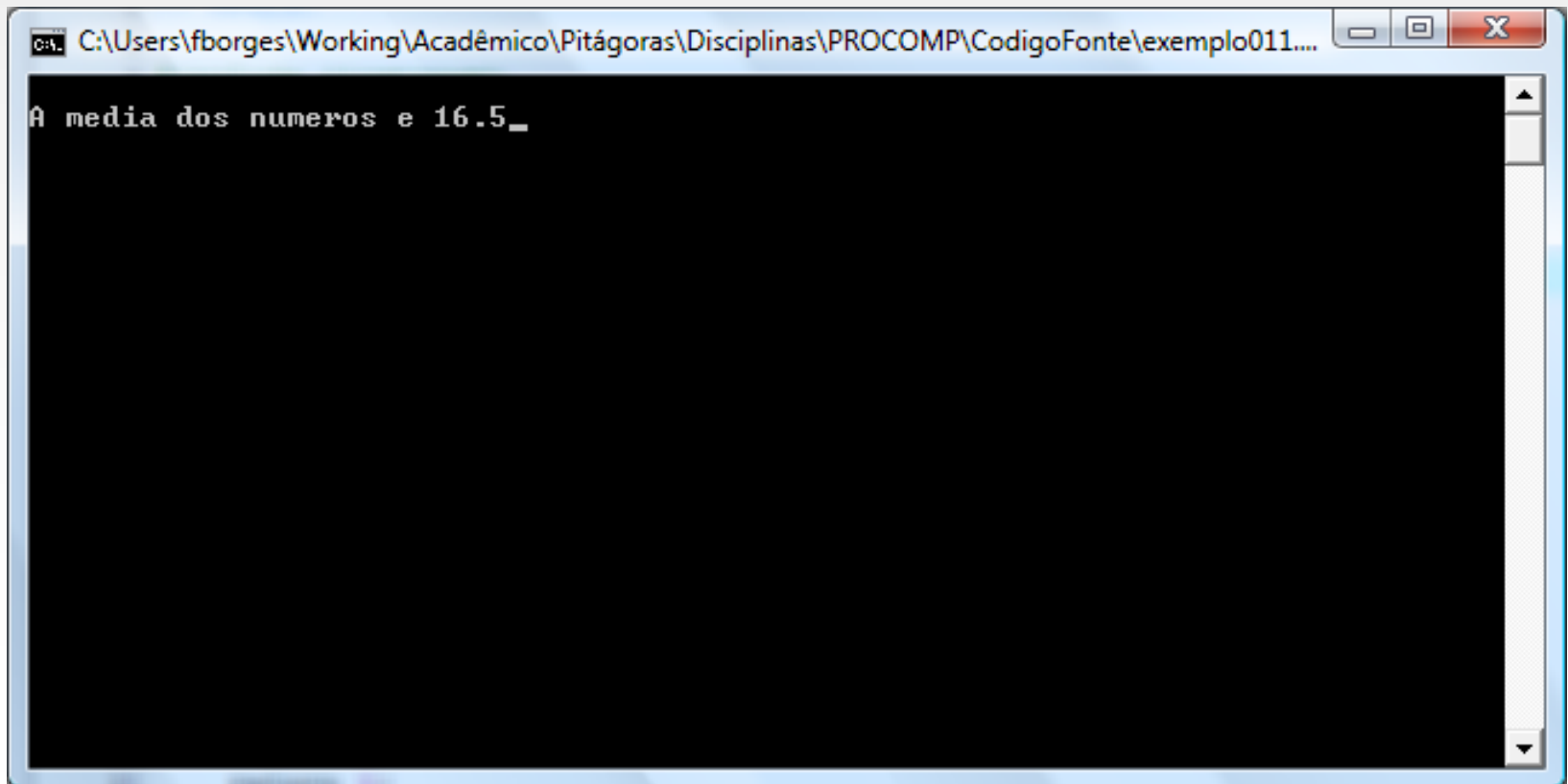
int main(){
    int numeros[2][4] = {{10, 11, 12, 13}, {20, 21, 22, 23}};
    double total = 0, qtde = 0;

    for (int ctLin = 0; ctLin < 2; ctLin++){
        for (int ctCol = 0; ctCol < 4; ctCol++){
            qtde++;
            total += numeros[ctLin][ctCol];
        }
    }

    printf("\nA media dos numeros e %.2f", total/qtde);
    return 0;
}
```

Vetores Multidimensionais

- Resultado da execução do programa



A screenshot of a Windows command prompt window. The title bar at the top shows the file path: `C:\Users\fborges\Working\Acadêmico\Pitágoras\Disciplinas\PROCOMP\CodigoFonte\exemplo011....`. The window has standard Windows window controls (minimize, maximize, close) on the right. The main area is black with white text. The text displayed is `A media dos numeros e 16.5_` followed by a cursor. There is a vertical scrollbar on the right side of the window.

Vetores Multidimensionais

■ Exemplo 3

- Uma empresa de construção deseja adquirir alguns materiais para uma obra (areia, tijolo e cimento) em um único fornecedor, o de menor custo. Sabendo que o preço de cada material corresponde a uma coluna de um vetor bidimensional e as linhas representam os fornecedores, que são quatro, escreva um programa em C que permita ao usuário informar os preços de cada material em cada fornecedor e que exiba, ao final de sua execução, o fornecedor cujos materiais possuem menor custo.

Vetores Multidimensionais

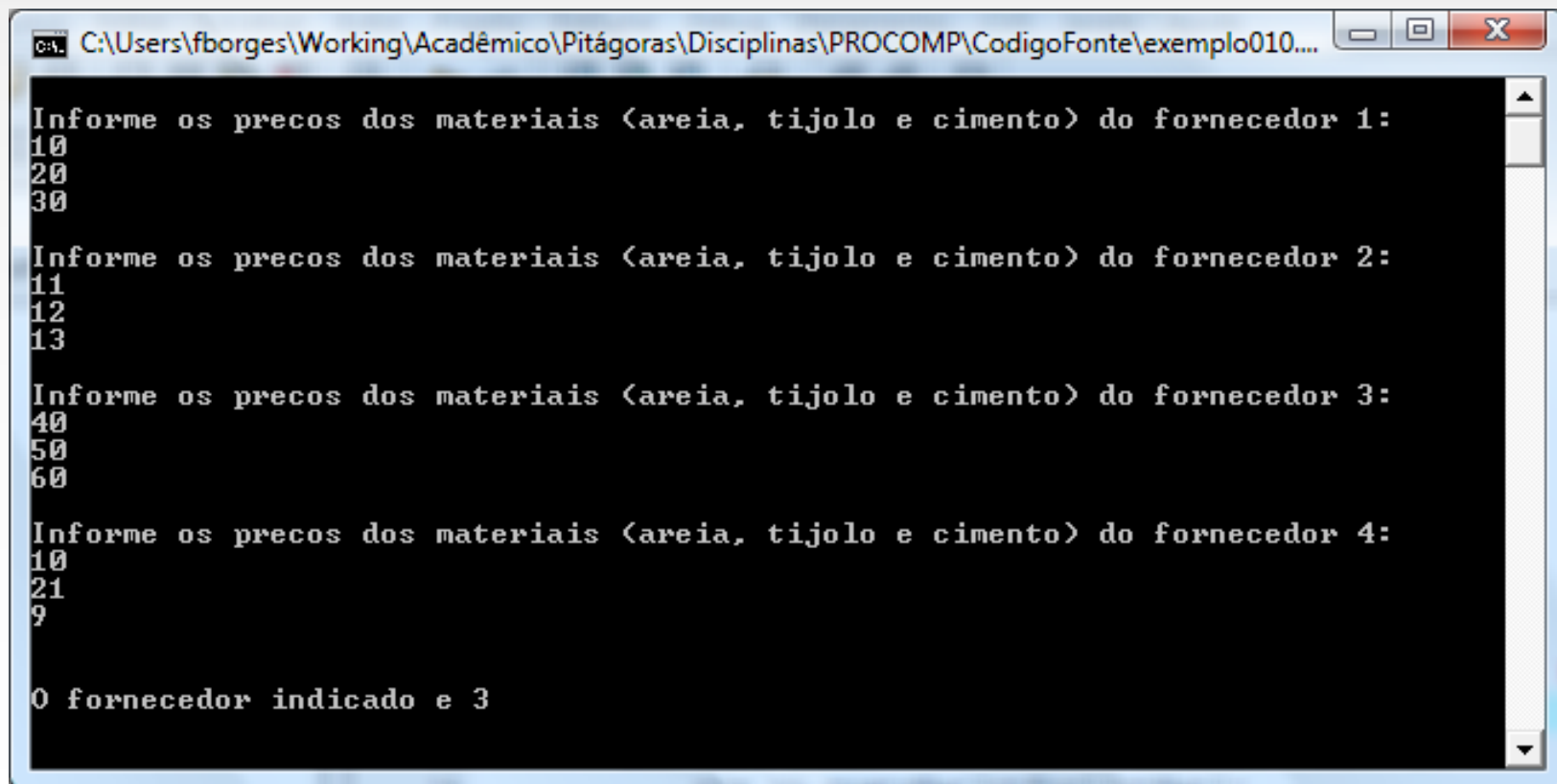
```
#include <stdio.h>
int main()
{
    const int TOTFOR = 4; const int TOTMAT = 3;
    float custoMat[TOTFOR][TOTMAT];
    for (int ctFor = 0; ctFor < TOTFOR; ctFor++) {
        printf("\nInforme os precos dos materiais ")
        printf("(areia, tijolo e cimento)");
        printf(" do fornecedor %d:\n", ctFor + 1);
        for (int ctMat = 0; ctMat < TOTMAT; ctMat++) {
            scanf("%f", &custoMat[ctFor][ctMat]);
        }
    }
}
```


Vetores Multidimensionais

```
int idFor = -1, menorCusto = 9999;
float custoFor = 0;
for (int ctFor = 0; ctFor < TOTFOR; ctFor++) {
    custoFor = 0;
    for (int ctMat = 0; ctMat < TOTMAT; ctMat++) {
        custoFor += custoMat[ctFor][ctMat];
    }
    if (custoFor < menorCusto) {
        idFor = ctFor;
        menorCusto = custoFor;
    }
}
printf("\nO fornecedor indicado e %d.", idFor+1);
return 0;
}
```

Vetores Multidimensionais

- Resultado da execução do programa



```
C:\Users\fborges\Working\Acadêmico\Pitágoras\Disciplinas\PROCOMP\CodigoFonte\exemplo010....  
Informe os precos dos materiais <areia, tijolo e cimento> do fornecedor 1:  
10  
20  
30  
Informe os precos dos materiais <areia, tijolo e cimento> do fornecedor 2:  
11  
12  
13  
Informe os precos dos materiais <areia, tijolo e cimento> do fornecedor 3:  
40  
50  
60  
Informe os precos dos materiais <areia, tijolo e cimento> do fornecedor 4:  
10  
21  
9  
  
O fornecedor indicado e 3
```

Cadeia de Caracteres

- O C não possui um tipo de dados específico para armazenamento de uma cadeia de caracteres
- Diante disso, utiliza-se vetores unidimensionais onde cada posição de um vetor é um caractere (***char***)
- Sempre a última posição do vetor para uma cadeia de caracteres é marcada com nulo, sendo assim, um vetor de 10 posições armazena uma cadeia de 9 caracteres

Cadeia de Caracteres

■ Exemplo 1

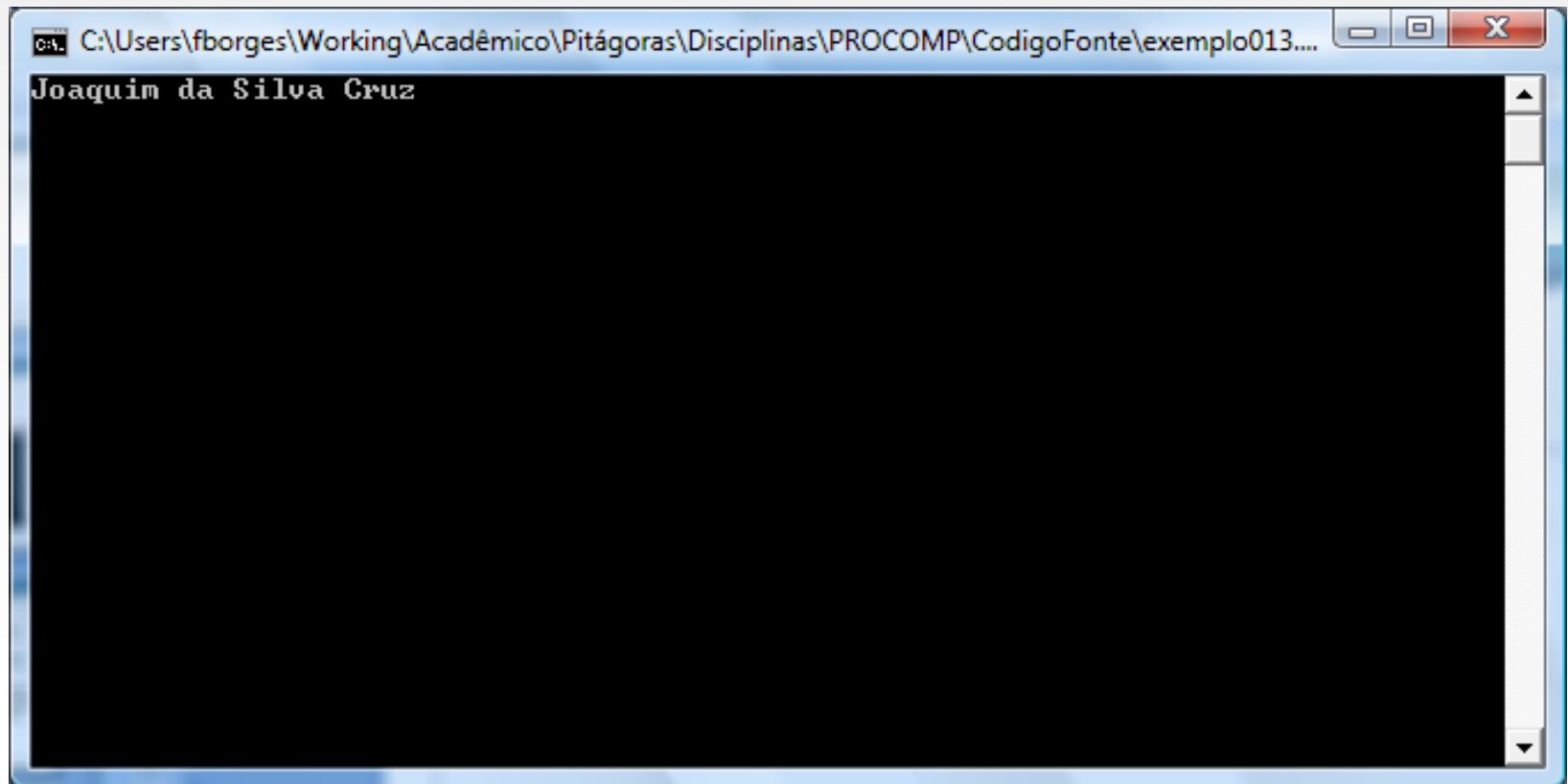
- Escreva um programa que, por meio de uma cadeia de caracteres, inicialize e exiba o nome completo de uma pessoa

```
#include <stdio.h>

int main()
{
    char nome[80] = "Joaquim da Silva Cruz";
    printf("%s", nome);
    printf("\n%.7s", nome);
    return 0;
}
```

Cadeia de Caracteres

- Resultado da execução do programa



A screenshot of a Windows command prompt window. The title bar at the top shows the file path: `C:\Users\fborges\Working\Acadêmico\Pitágoras\Disciplinas\PROCOMP\CodigoFonte\exemplo013....`. The window has standard Windows window controls (minimize, maximize, close) on the right. The main area of the window is black, and the text `Joaquim da Silva Cruz` is displayed in a white, monospaced font at the top left of the command area.

Cadeia de Caracteres

■ Exemplo 2

- Escreva um programa que preencha um vetor unidimensional de caracteres com o nome digitado pelo usuário e exiba-o

Cadeia de Caracteres

```
#include <stdio.h>
int main()
{
    char nome[80];

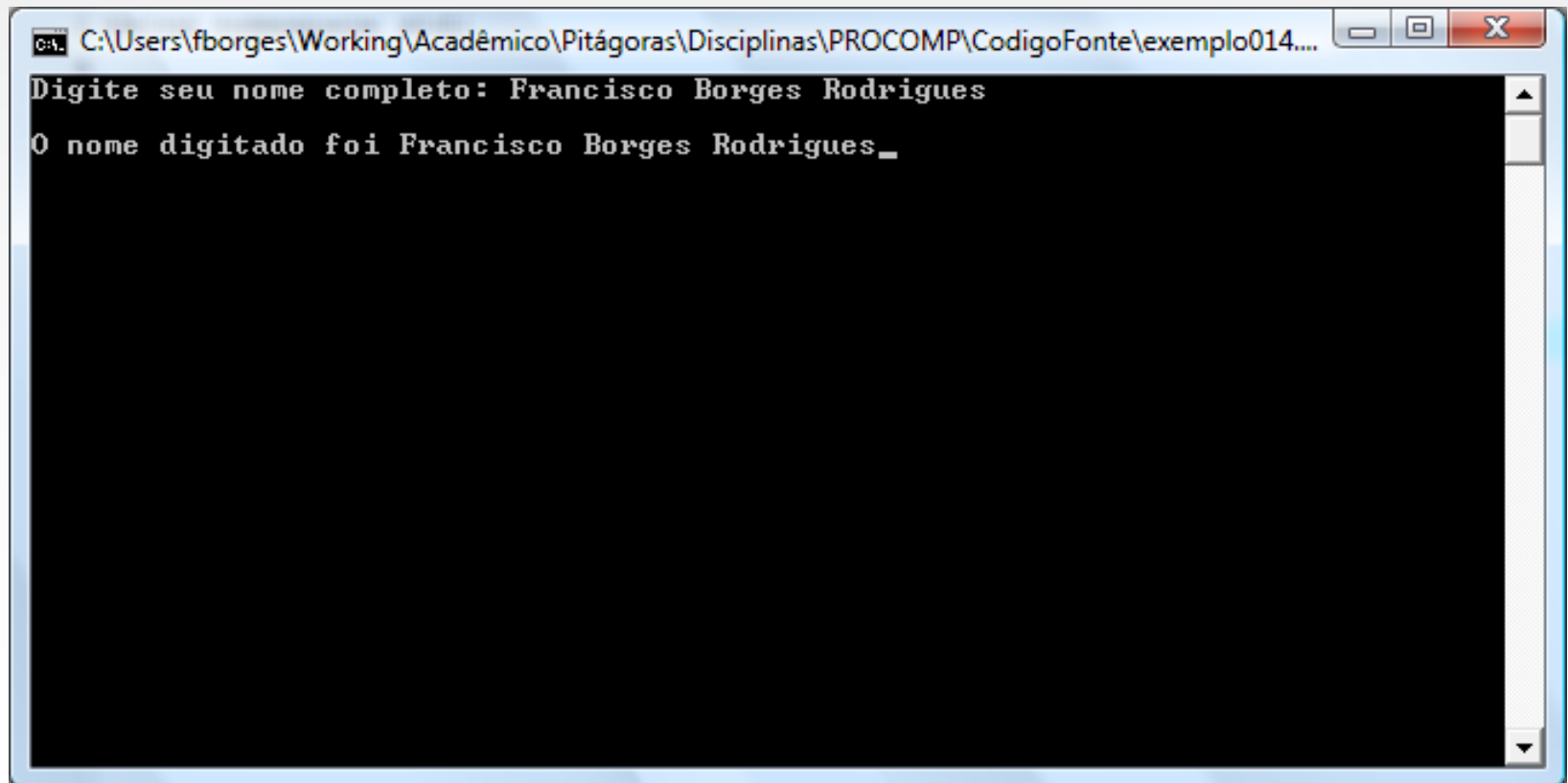
    printf("Digite seu nome completo:");
    scanf("%s", &nome);

    printf("O nome digitado foi: %s", nome);

    return 0;
}
```

Cadeia de Caracteres

- Resultado da execução do programa



```
C:\Users\fborges\Working\Acadêmico\Pitágoras\Disciplinas\PROCOMP\CodigoFonte\exemplo014....  
Digite seu nome completo: Francisco Borges Rodrigues  
O nome digitado foi Francisco Borges Rodrigues_
```


Cadeia de Caracteres

■ Exemplo 3

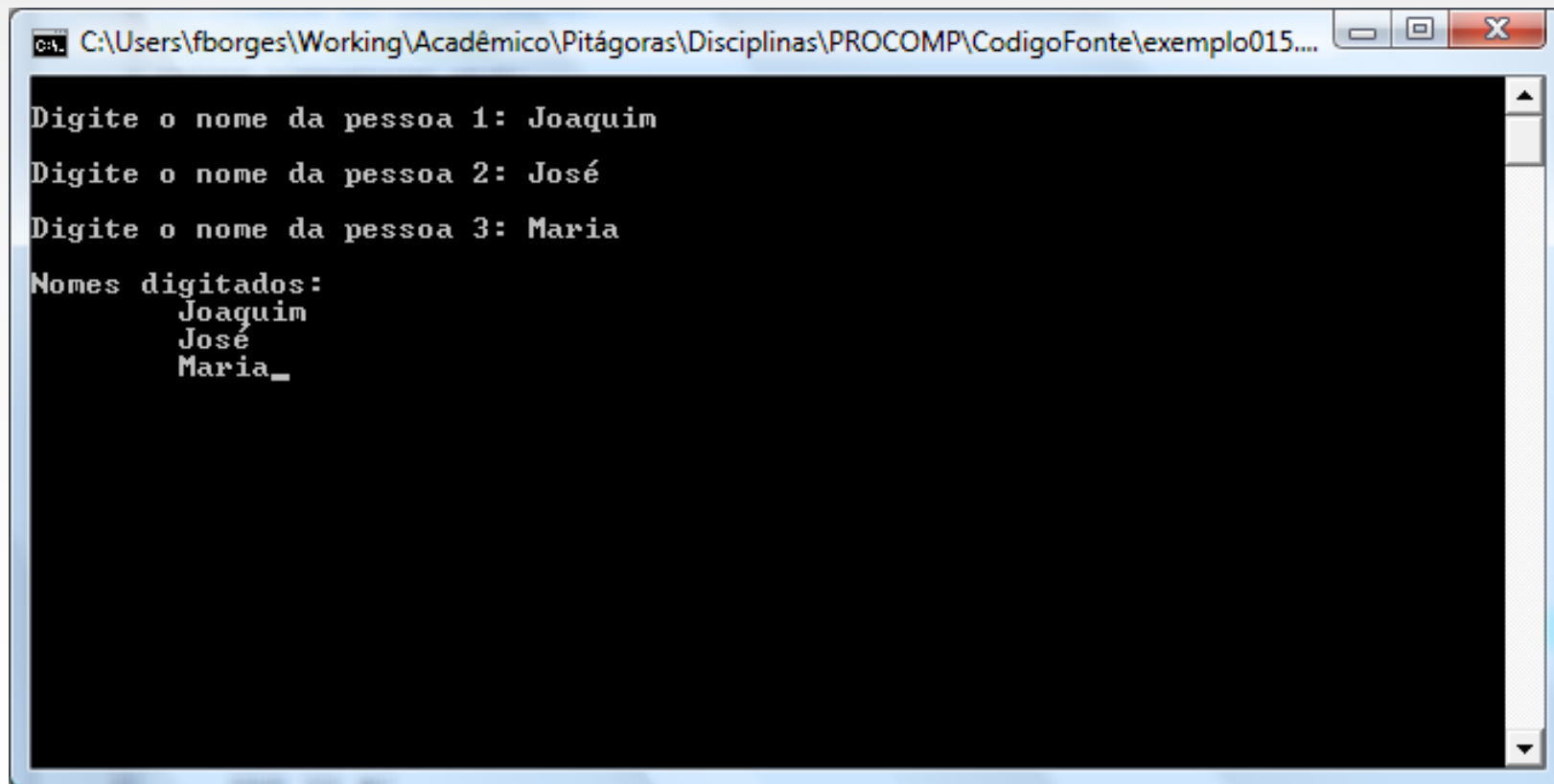
- Escreva um programa que armazene em uma matriz os nomes de três pessoas informados pelo usuário e ao final de sua execução mostre os três nomes digitados

Cadeia de Caracteres

```
#include <stdio.h>
int main()
{
    const int TOTPES = 3;
    char nome[TOTPES][80];
    for (int i = 0; i < TOTPES; i++) {
        printf("\nDigite o nome da pessoa %d :", i+1);
        scanf(" %79[^\n]s", nome[i]);
    }
    printf("\nNomes digitados: ");
    for (int i = 0; i < TOTPES; i++) {
        printf("\n\t%s", nome[i]);
    }
    return 0;
}
```

Cadeia de Caracteres

- Resultado da execução do programa



```
C:\Users\fborges\Working\Acadêmico\Pitágoras\Disciplinas\PROCOMP\CodigoFonte\exemplo015....  
Digite o nome da pessoa 1: Joaquim  
Digite o nome da pessoa 2: José  
Digite o nome da pessoa 3: Maria  
Nomes digitados:  
    Joaquim  
    José  
    Maria_
```

Modularização

- A maioria dos programas resolvem problemas do mundo real e são bem maiores que os apresentados até aqui
- A melhor maneira de desenvolver e manter um programa grande é construí-lo a partir de pequenas e simples partes
- Essas partes constituem módulos independentes que realizam pequenas tarefas
- Essa técnica é conhecida como **dividir para conquistar**

Modularização

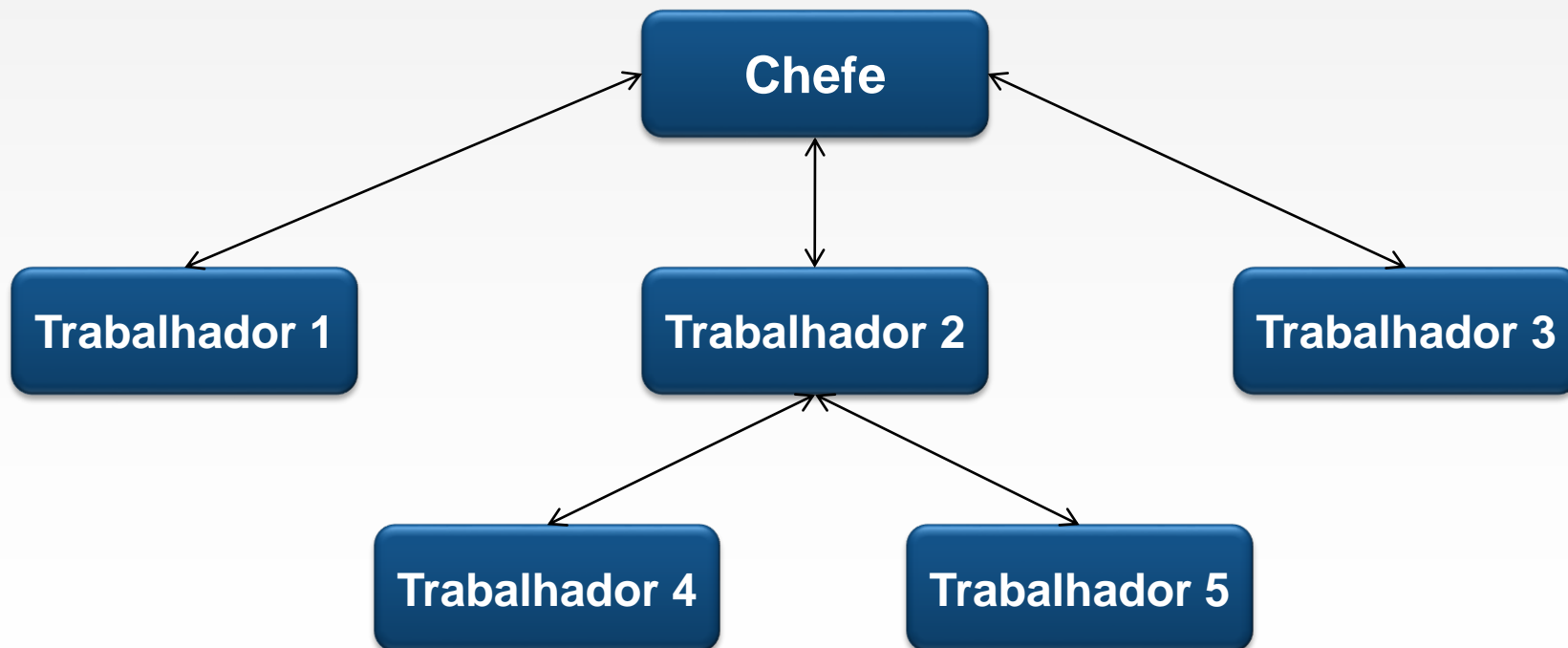
- Módulos de programas são utilizados para:
 - Permitir o reaproveitamento do código;
 - Evitar redundância de código;
 - Permitir uma manutenção mais eficiente e barata;
 - Tornar o código fonte mais legível; e,
 - Dividir o programa em partes que possam ser logicamente compreendidas de forma isolada.

Modularização

- Os módulos são conhecidos nas linguagens de programação como ***métodos***, ***procedimentos*** ou ***funções***
- Em C a modularização é realizada por meio de ***funções***
- Uma função é invocada por uma chamada de função e, quando a função chamada completa sua tarefa, ela retorna um resultado, ou simplesmente o controle, para o chamador

Modularização

- Pode-se fazer uma analogia à forma hierárquica de gerenciamento



Funções

- As funções utilizadas em um programa em C se dividem em:
 - Funções pré-definidas: funções presentes nas bibliotecas do próprio C; e,
 - Funções definidas pelo programador: funções implementadas pelos programadores, quando necessário.

Funções Pré-definidas

- A utilização das funções pré-definidas dependem da inclusão do arquivo de cabeçalho
- Abaixo algumas funções do **<cmath>**

Função	Descrição	Exemplo
<code>pow (x, y)</code>	x elevado à potência y (x^y)	<code>pow (2, 3)</code> é 8
<code>sqrt (x)</code>	Raiz quadrada de x (onde x é positivo)	<code>sqrt (4)</code> é 2
<code>ceil (x)</code>	Arredonda x para o menor inteiro maior que x	<code>ceil (9.2)</code> é 10
<code>floor (x)</code>	Arredonda x para o maior inteiro menor que x	<code>floor (9.7)</code> é 9
<code>cos (x)</code>	Co-seno trigonométrico de x	<code>cos (0.0)</code> é 1.0
<code>sin (x)</code>	Seno trigonométrico de x	<code>sin (0.0)</code> é 0.0

Funções Pré-definidas

■ Exemplo 1

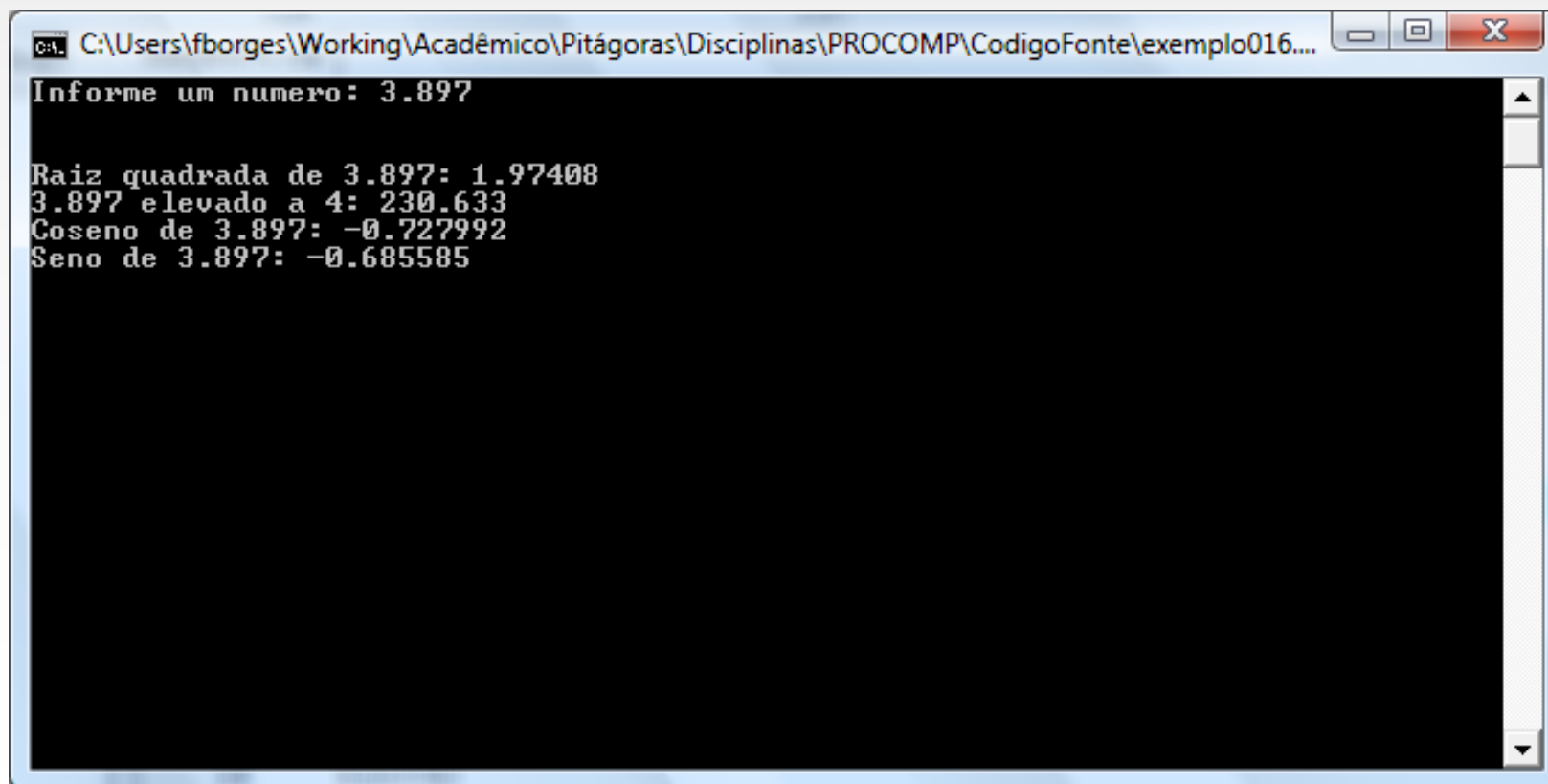
- Escreva um programa em C que receba um número digitado pelo usuário e exiba o resultado das seguintes operações:
 - Raiz quadrada do número
 - Número elevado à potência 4
 - Co-seno
 - Seno

Funções Pré-definidas

```
#include <stdio.h>
#include <math.h>
int main()
{
    float num;
    printf("Informe um numero:");
    scanf("%f", &num);
    printf("\nRaiz quadrada de %.2f: %.2f", num,
          sqrt(num));
    printf("\n%d elevado a 4: %g", num, pow(num, 4));
    printf("\nCoseno de %.2f : %g", num, cos(num));
    printf("\nSeno de %.2f : %g", num, sin(num));
    return 0;
}
```

Funções Pré-definidas

- Resultado da execução do programa



```
C:\Users\fborges\Working\Acadêmico\Pitágoras\Disciplinas\PROCOMP\CodigoFonte\exemplo016....  
Informe um numero: 3.897  
  
Raiz quadrada de 3.897: 1.97408  
3.897 elevado a 4: 230.633  
Coseno de 3.897: -0.727992  
Seno de 3.897: -0.685585
```

Funções Pré-definidas

■ Exemplo 2

- Dado uma matriz 5 x 5 de números reais, calcule e exiba a raiz de cada elemento.

Funções Pré-definidas

```
#include <stdio.h>
#include <math.h>

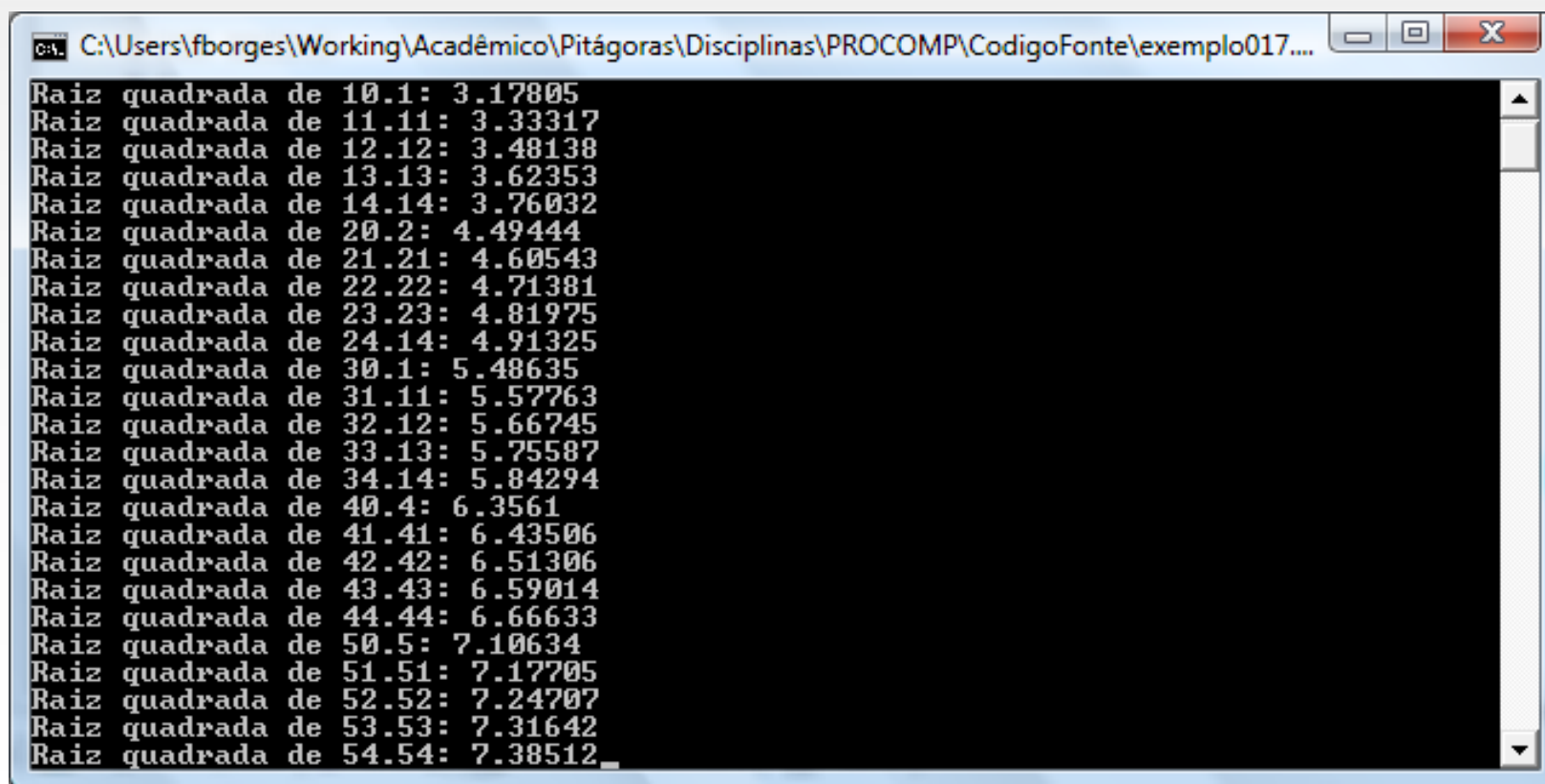
int main()
{
    float nums[5][5] = {{10.10, 11.11, 12.12, 13.13, 14.14},
                        {20.20, 21.21, 22.22, 23.23, 24.14},
                        {30.10, 31.11, 32.12, 33.13, 34.14},
                        {40.40, 41.41, 42.42, 43.43, 44.44},
                        {50.50, 51.51, 52.52, 53.53, 54.54}};

    for (int i = 0; i < 5; i++)
        for (int j = 0; j < 5; j++)
            printf("\nRaiz quadrada de %.2f: %.2f",
                  nums[i][j], sqrt(nums[i][j]));

    return 0;
}
```

Funções Pré-definidas

- Resultado da execução do programa



```
C:\Users\fborges\Working\Acadêmico\Pitágoras\Disciplinas\PROCOMP\CodigoFonte\exemplo017....  
Raiz quadrada de 10.1: 3.17805  
Raiz quadrada de 11.11: 3.33317  
Raiz quadrada de 12.12: 3.48138  
Raiz quadrada de 13.13: 3.62353  
Raiz quadrada de 14.14: 3.76032  
Raiz quadrada de 20.2: 4.49444  
Raiz quadrada de 21.21: 4.60543  
Raiz quadrada de 22.22: 4.71381  
Raiz quadrada de 23.23: 4.81975  
Raiz quadrada de 24.14: 4.91325  
Raiz quadrada de 30.1: 5.48635  
Raiz quadrada de 31.11: 5.57763  
Raiz quadrada de 32.12: 5.66745  
Raiz quadrada de 33.13: 5.75587  
Raiz quadrada de 34.14: 5.84294  
Raiz quadrada de 40.4: 6.3561  
Raiz quadrada de 41.41: 6.43506  
Raiz quadrada de 42.42: 6.51306  
Raiz quadrada de 43.43: 6.59014  
Raiz quadrada de 44.44: 6.66633  
Raiz quadrada de 50.5: 7.10634  
Raiz quadrada de 51.51: 7.17705  
Raiz quadrada de 52.52: 7.24707  
Raiz quadrada de 53.53: 7.31642  
Raiz quadrada de 54.54: 7.38512
```

Funções Definidas pelo Prog.

- O programador pode definir funções quando necessário
 - Essas funções podem ser definidas no próprio programa ou em arquivos separados para serem utilizadas em diversos programas
 - Segue abaixo sintaxe para definição de uma função
- <tipo_retorno> <nome_função> ([lista_de_parâmetros])**

Funções Definidas pelo Prog.

- Pode-se dividir as funções definidas pelo programador em:
 - Funções que retornam valor: são funções que executam uma tarefa específica e retornam um valor resultante da tarefa executada; e,
 - Funções que não retornam valor: também conhecidas como ***funções void***. São funções que simplesmente executam uma tarefa sem a necessidade de retornar um valor ao chamador.
- A seguir um exemplo de função que retorna valor

Funções Definidas pelo Prog.

■ Calcular o somatório de um vetor unidimensional

```
#include <stdio.h>
int main() {

    int num[4] = {1, 2, 3, 4};
    int total = 0;

    for (int i = 0; i < 4; i++)
        total += num[i];

    cout << total;

    return 0;
}
```

```
#include <stdio.h>

int calcTot(int vet[], int tam);

int main() {
    int num[4] = {1, 2, 3, 4};
    printf("%2.f",
           calcTot(num, 4));
    return 0;
}
```

```
int calcTot(int vet[], int tam){
    int total = 0;
    for (int i = 0; i < tam; i++)
        total += vetor[i];
    return total;
}
```

Passagem de Parâmetros

- Existem duas maneiras de passar argumentos como parâmetros:
 - Passagem por valor
 - Uma cópia do valor do argumento é passada para função
 - O parâmetro é considerado uma variável local dentro da função
 - As alterações na cópia não afetam o valor original no chamador
 - Se modificado, afeta somente a cópia local
 - Método padrão utilizado
 - **Não** é necessário declarar o parâmetro novamente dentro da função como uma variável local!

Passagem de Parâmetros

■ Exemplo passagem de parâmetro por valor

```
#include <stdio.h>

int soma(int num1, int num2);

int main(){
    int num1, num2;
    printf("Informe dois numeros:");
    scanf("%d %d", &num1, &num2);
    printf("\nO resultado da soma desses numeros e: %d",
        soma(num1, num2));
    return 0;
}

int soma(int num1, int num2){
    return num1 + num2;
}
```

Passagem de Parâmetros

- Passagem por referência
 - O **endereço** atual do argumento é passado
 - A função chamada têm acesso direto ao valor do argumento
 - É um **alias** (apelido) para um argumento
 - Permite a modificação do valor do argumento utilizado pelo chamador
 - O “**e**” comercial (**&**) é utilizado para indicar tal parâmetro

Passagem de Parâmetros

■ Exemplo passagem de parâmetro por referência

```
#include <stdio.h>
void soma(int *num1, int num2);
int main(){
    int num1=0, num2=0;
    printf("Informe dois numeros:");
    scanf("%i %i", &num1, &num2);
    soma(&num1, num2);
    printf("\nO resultado da soma desses numeros e: %i",
        num1);
    return 0;
}

void soma(int *num1, int num2){
    *num1 += num2;
}
```

Exercícios

1. Uma empresa transportadora deseja controlar os gastos com combustível da sua frota de 200 veículos. Escreva um programa em C++ que permita ao usuário informar o gasto mensal de cada um dos 200 veículos e, a partir desses dados, exiba as seguintes informações, retornadas por funções:
 - Gasto médio
 - Maior gasto
 - Menor gasto

Exercícios

2. Uma empresa de construção possui o currículo com pretensão salarial de nove candidatos a pedreiro. Ela precisa contratar cinco pedreiros para a execução de uma construção. Sabendo que os salários pretendidos estão armazenados em um vetor unidimensional, faça um programa que retorne a identificação dos cinco pedreiros com menores pretensões salariais.

0 – R\$ 984,59	1 – R\$ 876,77	2 – R\$ 867,56
3 – R\$ 858,90	4 – R\$ 845,99	5 – R\$ 768,98
6 – R\$ 678,88	7 – R\$ 803,05	8 – R\$ 765,90

Exercícios

3. Em uma pesquisa estatística realizada em uma maternidade, foram coletados o peso em gramas e a semana de gestação de cada um dos 20 bebês que lá nasceram. Escreva um programa em C que leia o peso e a semana de gestação do nascimento de cada um dos 20 bebês e realize as seguintes tarefas:
 - a) Calcular a quantidade de bebês prematuros (antes de 37 semanas) nascidos com baixo peso (abaixo de 2.500 gr.);
 - b) Calcular a quantidade de bebês prematuros nascidos com peso bom (a partir de 2.500 gr.);
 - c) Calcular o peso médio dos bebês prematuros;
 - d) Exibir um relatório com o peso e a semana de gestação do nascimento de cada bebê e ao final os dados calculados nos itens anteriores.

Exercícios

4. Escreva em C++ uma função que receba dois números inteiros como parâmetros e realize a subtração do primeiro com o segundo, armazenando o resultado dessa subtração no primeiro parâmetro. A função deverá usar, pelo menos, uma passagem de parâmetro por referência