MC102 – Algoritmos e Programação de Computadores

Instituto de Computação

UNICAMP

Segundo Semestre de 2013

Roteiro

Vetores

Como armazenar 3 notas?

```
float nota1, nota2, nota3;

printf("Nota do aluno 1: ");
scanf("%f", &nota1);
printf("Nota do aluno 2: ");
scanf("%f", &nota2);
printf("Nota do aluno 3: ");
scanf("%f", &nota3);
```

Vetores

Como armazenar 100 notas?

```
float nota1, nota2, nota3, /* .... */ nota100;

printf("Nota do aluno 1: ");
scanf("%f", &nota1);
printf("Nota do aluno 2: ");
scanf("%f", &nota2);

/* ... */

printf("Nota do aluno 100: ");
scanf("%f", &nota100);
```

Vetores

Definição: coleção de variáveis do mesmo tipo referenciada por um nome comum.

- Acesso por meio de índice inteiro.
- Posições contíguas na memória.
- Tamanho pré-definido.
- Índices fora dos limites podem causar comportamento anômalo do programa.

Declaração de um vetor

```
<tipo> identificador[<tamanho do vetor>];
```

Exemplos:

```
float notas[100];
int medias[100];
char nome[200];
```

Usando um vetor

Após declarada uma variável do tipo vetor, pode-se ter acesso a uma determinada posição do vetor utilizando um valor inteiro.

```
identificador[<posição>]
```

- O acesso de um vetor em uma posição específica tem o mesmo comportamento que uma variável simples.
- Em C, a primeira posição de um vetor tem índice 0.
- A última posição de um vetor tem índice `<tamanho do vetor> - 1`.

Exemplo:

```
int nota[10];
int a;
nota[5] = 95;
a = nota[5];
```

Usando um vetor

- Pode-se usar valores inteiros, tanto constantes quanto variáveis, para acessar uma posição do vetor.

Exemplo:

```
int i, v[10];
for (i = 0; i < 10; i++)
  v[i] = 10 * i;
```

Vetores

- Declaração de variáveis:

```
int a;
int vetor[5];
int b;
```

Nome	a	vetor					b
Índice	-	0	1	2	3	4	-
Valor							

Vetores

- Acessos válidos:

```
a = 3;
vetor[a] = 2;
vetor[1] = vetor[a] + 5;
```

Nome	a	vetor					b
Índice	-	0	1	2	3	4	-
Valor	3		7		2		

Vetores

- Acessos inválidos:

```
vetor[5] = 4;  // Erro: alterou o valor de b
vetor[-1] = 6; // Erro: alterou o valor de a
vetor[10] = 8; // Erro: segmentation fault
```

Nome	a	vetor					b
Índice	-	0	1	2	3	4	-
Valor	6		7		2		4

Questões importantes sobre vetores

- O tamanho do vetor é pré-definido, ou seja, não pode ser alterado durante a execução do programa.
- Índices fora dos limites podem causar comportamento anômalo do código.

Como armazenar n (≤ 100) notas?

```
#include <stdio.h>

int main() {
  float nota[100];
  int n, i;

  printf("Numero de alunos: ");
  scanf("%d", &n);

  for (i = 0; i < n; i++) {
    printf("Nota do aluno %d: ", i+1);
    scanf("%f", &nota[i]);
  }

  return 0;
}
```

- Pode ocorrer algum problema durante a execução do programa?

Como armazenar n (≤ 100) notas?

```
#include <stdio.h>

int main() {
  float nota[100];
  int n, i;

  do {
    printf("Numero de alunos: ");
    scanf("%d", &n);
  } while ((n < 0) || (n > 100));

  for (i = 0; i < n; i++) {
    printf("Nota do aluno %d: ", i+1);
    scanf("%f", &nota[i]);
  }

  return 0;
}
```

Média de valores

- Ler um vetor com 10 valores inteiros e computar a média dos valores.
- Quais tipos de variáveis usar?

Média de valores

```
#include <stdio.h>

int main() {
  int i, valores[10], soma;

  for (i = 0; i < 10; i++) {
    printf("Digite valor %d: ", i+1);
    scanf("%d", &valores[i]);
  }

  soma = 0;

  for (i = 0; i < 10; i++)
    soma += valores[i];

  printf("Media: %.1f\n", soma / 10.0);

  return 0;
}
```

Média de valores

```
#include <stdio.h>

int main() {
  int i, valores[10], soma;

  for (i = 0; i < 10; i++) {
    printf("Digite valor %d: ", i+1);
    scanf("%d", &valores[i]);
  }

  soma = 0;

  for (i = 0; i < 10; i++)
    soma += valores[i];

  printf("Media: %.1f\n", soma / 10.0);

  return 0;
}
```

Produto interno de dois vetores

- Ler dois vetores de dimensão 5 e computar o produto interno destes.
- Quais tipos de variáveis usar?

Produto interno de dois vetores

```
#include <stdio.h>

int main() {
  double vetor1[5], vetor2[5], resultado;
  int i;

  for (i = 0; i < 5; i++) {
    printf("Entre com valor %d para vetor 1: ", i+1);
    scanf("%lf", &vetor1[i]);
  }

  for (i = 0; i < 5; i++) {
    printf("Entre com valor %d para vetor 2: ", i+1);
    scanf("%lf", &vetor2[i]);
  }

  resultado = 0.0;
  for (i = 0; i < 5; i++)
    resultado = resultado + (vetor1[i] * vetor2[i]);

  printf("Produto interno: %f\n", resultado);
  return 0;
}
```

Produto interno de dois vetores

```
#include <stdio.h>

int main() {
  double vetor1[5], vetor2[5], resultado;
  int i;

  for (i = 0; i < 5; i++) {
    printf("Entre com valor %d para vetor 1: ", i+1);
    scanf("%lf", &vetor1[i]);
  }

  for (i = 0; i < 5; i++) {
    printf("Entre com valor %d para vetor 2: ", i+1);
    scanf("%lf", &vetor2[i]);
  }

  resultado = 0.0;
  for (i = 0; i < 5; i++)
    resultado = resultado + (vetor1[i] * vetor2[i]);

  printf("Produto interno: %f\n", resultado);
  return 0;
}
```

Elementos iguais

- Ler dois vetores com 5 inteiros cada.
- Identificar quais elementos do segundo vetor são iguais a algum elemento do primeiro vetor.

Elementos iguais

```
#include <stdio.h>

int main() {
  int vetor1[5], vetor2[5], i, j;

  for (i = 0; i < 5; i++) {
    printf("Entre com o valor de vetor1[%d]: ", i);
    scanf("%d", &vetor1[i]);
  }

  for (i = 0; i < 5; i++) {
    printf("Entre com o valor de vetor2[%d]: ", i);
    scanf("%d", &vetor2[i]);
  }

  for (i = 0; i < 5 ; i++)
    for (j = 0; j < 5; j++)
      if (vetor1[i] == vetor2[j])
        printf("vetor1[%d] = vetor2[%d] = %d\n", i, j, vetor1[i]);

  return 0;
}
```

Elementos iguais

```
#include <stdio.h>

int main() {
  int vetor1[5], vetor2[5], i, j;

  for (i = 0; i < 5; i++) {
    printf("Entre com o valor de vetor1[%d]: ", i);
    scanf("%d", &vetor1[i]);
  }

  for (i = 0; i < 5; i++) {
    printf("Entre com o valor de vetor2[%d]: ", i);
    scanf("%d", &vetor2[i]);
  }

  for (i = 0; i < 5 ; i++)
    for (j = 0; j < 5; j++)
      if (vetor1[i] == vetor2[j])
        printf("vetor1[%d] = vetor2[%d] = %d\n", i, j, vetor1[i]);

  return 0;
}
```

Habemos Papam

- As eleições dos papas são realizadas por Conclaves, reuniões que reunem todos os cardeais com menos de 80 anos, onde todos são elegíveis e têm direito a voto.
- O Conclave de 2013 contou com a presença de 114 cardeais.
- Para um cardeal ser eleito papa, ele precisa obter mais de 2/3 dos votos, ou seja, 77 votos no caso do Conclave de 2013.
- Supondo que os cardeais foram numerados de 1 a 114 (e o número 0 foi usado para representar votos brancos ou nulos), dada a lista de votos, como determinar se um novo papa foi eleito?

Habemos Papam

```
#include <stdio.h>

int main() {
  int cardeal[115], n, i, voto, papa = 0;

  for (i = 0; i <= 114; i++)
    cardeal[i] = 0;

  for (i = 1; i <= 114; i++) {
    printf("Qual o cardeal indicado?\n");
    scanf("%d", &voto);
    cardeal[voto]++;
  }

  for (i = 1; i <= 114; i++)
    if (cardeal[i] > cardeal[papa])
      papa = i;

  if (papa && (cardeal[papa] >= 77))
    printf("Habemus Papam: cardeal %d\n", papa);

  return 0;
}
```

Habemos Papam

```
#include <stdio.h>

int main() {
  int cardeal[115], n, i, voto, papa = 0;

  for (i = 0; i <= 114; i++)
    cardeal[i] = 0;

  for (i = 1; i <= 114; i++) {
    printf("Qual o cardeal indicado?\n");
    scanf("%d", &voto);
    cardeal[voto]++;
  }

  for (i = 1; i <= 114; i++)
    if (cardeal[i] > cardeal[papa])
      papa = i;

  if (papa && (cardeal[papa] >= 77))
    printf("Habemus Papam: cardeal %d\n", papa);

  return 0;
}
```

Habemos Papam

```
#include <stdio.h>

int main() {
  int cardeal[115], n, i, voto, papa = 0;

  for (i = 0; i <= 114; i++)
    cardeal[i] = 0;

  for (i = 1; i <= 114; i++) {
    printf("Qual o cardeal indicado?\n");
    scanf("%d", &voto);
    cardeal[voto]++;
  }

  for (i = 1; i <= 114; i++)
    if (cardeal[i] > cardeal[papa])
      papa = i;

  if (papa && (cardeal[papa] >= 77))
    printf("Habemus Papam: cardeal %d\n", papa);

  return 0;
}
```

Habemos Papam

```
#include <stdio.h>

int main() {
  int cardeal[115], n, i, voto, papa = 0;

  for (i = 0; i <= 114; i++)
    cardeal[i] = 0;

  for (i = 1; i <= 114; i++) {
    printf("Qual o cardeal indicado?\n");
    scanf("%d", &voto);
    cardeal[voto]++;
  }

  for (i = 1; i <= 114; i++)
    if (cardeal[i] >= 77)
      papa = i;

  if (papa)
    printf("Habemus Papam: cardeal %d\n", papa);

  return 0;
}
```

Strings

- A linguagem C não possui explicitamente o tipo `string`, entretanto, pode-se considerar um vetor de caracteres como uma string.
- Em C, uma string é sempre terminada pelo caractere especial '\0' (equivalente ao número zero).
- Portanto, sempre declare uma string com um caractere a mais do que você pretende usar.
- Se, por exemplo, pretender usar uma string de 10 caracteres, declare:

```
char string[11];
```

Strings

- Como já vimos, para ler ou imprimir uma string, usa-se o operador especial %s.

```
#include <stdio.h>

int main() {
  char palavra[81];
  int numero;

  printf("Entre com uma palavra: ");
  scanf("%s", palavra);
  printf("Entre com um numero: ");
  scanf("%d", &numero);
  printf("Palavra = %s\n", palavra);
  printf("Numero = %d\n", numero);

  return 0;
}
```

- Note que, para strings, não é utilizado o símbolo & no comando `scanf`.

Invertendo uma string

- Ler uma palavra de até 80 caracteres e salvar a inversa desta em um vetor (string).
- Imprimir a inversa da palavra lida.

Invertendo uma string

```
#include <stdio.h>

int main() {
  char palavra[81], inversa[81];
  int i, tam = 0;

  printf("Digite uma palavra: ");
  scanf("%s", palavra);

  while (palavra[tam] != '\0')
    tam++;

  inversa[tam] = '\0';
  for (i = 0; i < tam; i++)
    inversa[tam - i - 1] = palavra[i];

  printf("Inversa: %s\n", inversa);
  return 0;
}
```

Invertendo uma string

```
#include <stdio.h>

int main() {
  char palavra[81], inversa[81];
  int i, tam = 0;

  printf("Digite uma palavra: ");
  scanf("%s", palavra);

  while (palavra[tam] != '\0')
    tam++;

  inversa[tam] = '\0';
  for (i = 0; i < tam; i++)
    inversa[tam - i - 1] = palavra[i];

  printf("Inversa: %s\n", inversa);
  return 0;
}
```

Invertendo uma string

```
#include <stdio.h>

int main() {
  char palavra[81], inversa[81];
  int i, tam = 0;

  printf("Digite uma palavra: ");
  scanf("%s", palavra);

  while (palavra[tam] != '\0')
    tam++;

  inversa[tam] = '\0';
  for (i = 0; i < tam; i++)
    inversa[tam - i - 1] = palavra[i];

  printf("Inversa: %s\n", inversa);
  return 0;
}
```

Invertendo uma string

- Ler uma palavra de até 80 caracteres e salvar a inversa desta em um vetor (string).
- Imprimir a inversa da palavra lida.
- Usar apenas um vetor de caracteres.

Invertendo uma string

```
#include <stdio.h>

int main() {
  char palavra[81], temp;
  int i, tam = 0;

  printf("Digite uma palavra: ");
  scanf("%s", palavra);

  while (palavra[tam] != 0)
    tam++;

  for (i = 0; i < tam/2; i++) {
    temp = palavra[tam - i - 1];
    palavra[tam - i - 1] = palavra[i];
    palavra[i] = temp;
  }

  printf("Inversa: %s\n", palavra);
  return 0;
}
```

Invertendo uma string

```
#include <stdio.h>

int main() {
  char palavra[81], temp;
  int i, tam = 0;

  printf("Digite uma palavra: ");
  scanf("%s", palavra);

  while (palavra[tam] != 0)
    tam++;

  for (i = 0; i < tam/2; i++) {
    temp = palavra[tam - i - 1];
    palavra[tam - i - 1] = palavra[i];
    palavra[i] = temp;
  }

  printf("Inversa: %s\n", palavra);
  return 0;
}
```

Anagramas

- Um anagrama é o resultando do rearranjo dos caracteres de uma palavra ou frase para produzir outras palavras, utilizando todas os caracteres originais exatamente uma vez.
- Exemplo: “america” e “iracema”.
- Escreva um programa que leia duas palavras e verifique se elas são anagramas (uma em relação a outra).

Anagramas

```
#include <stdio.h>

int main() {
  char palavra1[21], palavra2[21], caracteres[128];
  int i, anagramas = 1;

  printf("Entre com duas palavras: ");
  scanf("%s %s", palavra1, palavra2);

  /* inicializa o vetor de frequencia de caracteres */
  for (i = 0; i < 128; i++)
    caracteres[i] = 0;

  /* conta os caracteres encontrados na palavra1 */
  for (i = 0; palavra1[i]; i++)
    caracteres[(int) palavra1[i]]++;

  /* conta os caracteres encontrados na palavra2 */
  for (i = 0; palavra2[i]; i++)
    caracteres[(int) palavra2[i]]--;
```

Anagramas

```
#include <stdio.h>

int main() {
  char palavra1[21], palavra2[21], caracteres[128];
  int i, anagramas = 1;

  printf("Entre com duas palavras: ");
  scanf("%s %s", palavra1, palavra2);

  /* inicializa o vetor de frequencia de caracteres */
  for (i = 0; i < 128; i++)
    caracteres[i] = 0;

  /* conta os caracteres encontrados na palavra1 */
  for (i = 0; palavra1[i]; i++)
    caracteres[(int) palavra1[i]]++;

  /* conta os caracteres encontrados na palavra2 */
  for (i = 0; palavra2[i]; i++)
    caracteres[(int) palavra2[i]]--;
```

Anagramas

```
#include <stdio.h>

int main() {
  char palavra1[21], palavra2[21], caracteres[128];
  int i, anagramas = 1;

  printf("Entre com duas palavras: ");
  scanf("%s %s", palavra1, palavra2);

  /* inicializa o vetor de frequencia de caracteres */
  for (i = 0; i < 128; i++)
    caracteres[i] = 0;

  /* conta os caracteres encontrados na palavra1 */
  for (i = 0; palavra1[i]; i++)
    caracteres[(int) palavra1[i]]++;

  /* conta os caracteres encontrados na palavra2 */
  for (i = 0; palavra2[i]; i++)
    caracteres[(int) palavra2[i]]--;
```

Anagramas

```
#include <stdio.h>

int main() {
  char palavra1[21], palavra2[21], caracteres[128];
  int i, anagramas = 1;

  printf("Entre com duas palavras: ");
  scanf("%s %s", palavra1, palavra2);

  /* inicializa o vetor de frequencia de caracteres */
  for (i = 0; i < 128; i++)
    caracteres[i] = 0;

  /* conta os caracteres encontrados na palavra1 */
  for (i = 0; palavra1[i]; i++)
    caracteres[(int) palavra1[i]]++;

  /* conta os caracteres encontrados na palavra2 */
  for (i = 0; palavra2[i]; i++)
    caracteres[(int) palavra2[i]]--;
```

Anagramas

```
  ...

  /* verifica se as palavras tem a mesma frequencia de caracteres */
  for (i = 0; i < 128; i++)
    if (caracteres[i] != 0)
      anagramas = 0;

  if (anagramas)
    printf("As duas palavras sao anagramas\n");
  else
    printf("As duas palavras nao sao anagramas\n");

  return 0;
}
```

Problemas com a leitura de cadeia de caracteres

```
#include <stdio.h>

int main() {
  char mensagem[21];

  printf("Digite uma mensagem: ");
  scanf("%s", mensagem);

  printf("Mensagem: %s\n", mensagem);

  return 0;
}
```

```
$ gcc -ansi -Wall -pedantic -Werror mensagem.c -o mensagem
$ ./mensagem
Digite uma mensagem: Bom dia, mundo!
Mensagem: Bom


$ ./mensagem
Digite uma mensagem: UmTesteComUmaMensagemMaisLongaPodeCausarUmErro
Mensagem: UmTesteComUmaMensagemMaisLongaPodeCausarUmErro
Segmentation fault
```

Problemas com a leitura de cadeia de caracteres

```
#include <stdio.h>

int main() {
  char mensagem[21];

  printf("Digite uma mensagem: ");
  scanf("%s", mensagem);

  printf("Mensagem: %s\n", mensagem);

  return 0;
}
```

```
$ gcc -ansi -Wall -pedantic -Werror mensagem.c -o mensagem
$ ./mensagem
Digite uma mensagem: Bom dia, mundo!
Mensagem: Bom
```

```
$ ./mensagem
Digite uma mensagem: UmTesteComUmaMensagemMaisLongaPodeCausarUmErro
Mensagem: UmTesteComUmaMensagemMaisLongaPodeCausarUmErro
Segmentation fault
```

Problemas com a leitura de cadeia de caracteres

```
#include <stdio.h>

int main() {
  char mensagem[21];

  printf("Digite uma mensagem: ");
  scanf("%s", mensagem);

  printf("Mensagem: %s\n", mensagem);

  return 0;
}
```

```
$ gcc -ansi -Wall -pedantic -Werror mensagem.c -o mensagem
$ ./mensagem
Digite uma mensagem: Bom dia, mundo!
Mensagem: Bom

$ ./mensagem
Digite uma mensagem: UmTesteComUmaMensagemMaisLongaPodeCausarUmErro
Mensagem: UmTesteComUmaMensagemMaisLongaPodeCausarUmErro
Segmentation fault
```

Comando fgets

- Ao usar o comando `scanf` para ler uma string, você deve garantir que foi alocada uma string de tamanho suficiente para armazenar todos os caracteres.
- Caso o programa tente ler mais caracteres do que o tamanho alocado, um erro ocorrerá durante a execução do programa.
- O comando `scanf` não é adequado para ler strings contendo espaços em branco.
- Uma alternativa para ler strings é o comando `fgets()`.

```
fgets(identificador, tamanho, stdin);
```

onde `identificador` é o nome da variável usada para armazenar a string e `tamanho` é um inteiro indicando até quantos caracteres devem ser lidos (até `tamanho-1` caracteres são lidos e um extra é reservado para o '\0').

Exemplo com fgets

```
#include <stdio.h>

int main() {
  char string[81], temp;
  int i, tam = 0;

  printf("Digite uma string: ");
  fgets(string, 81, stdin);

  while (string[tam] && (string[tam] != '\n'))
    tam++;

  for (i = 0; i < tam/2; i++) {
    temp = string[tam - i - 1];
    string[tam - i - 1] = string[i];
    string[i] = temp;
  }

  printf("Inversa: %s", string);
  return 0;
}
```

- Note que o comando `fgets` lê inclusive o caractere '\n'.

Cuidados com fgets

- No exemplo abaixo, a string não é lida corretamente. Por quê?

```
#include <stdio.h>

int main() {
  char string[81];
  int n;

  printf("Digite um numero: ");
  scanf("%d", &n);
  printf("Numero digitado: %d\n", n);

  printf("Digite um texto: ");
  fgets(string, 81, stdin);
  printf("Texto digitado: %s", string);

  return 0;
}
```

Cuidados com fgets

- O comando `scanf` não lê o caractere '\n' que representa o final de linha (*enter*).
- Este caractere fica armazenado no *buffer* da entrada padrão.
- A seguir, o comando `fgets` lê este caractere do *buffer*, o que automaticamente encerra a leitura da string.
- Como resolver este problema?
- Limpar o *buffer* da entrada padrão antes de usar o `fgets`, usando o comando:

```
setbuf(stdin, 0);
```

Cuidados com fgets

- O comando `scanf` não lê o caractere '\n' que representa o final de linha (*enter*).
- Este caractere fica armazenado no *buffer* da entrada padrão.
- A seguir, o comando `fgets` lê este caractere do *buffer*, o que automaticamente encerra a leitura da string.
- Como resolver este problema?
- Limpar o *buffer* da entrada padrão antes de usar o `fgets`, usando o comando:

```
setbuf(stdin, 0);
```

Cuidados com fgets

```
#include <stdio.h>

int main() {
  char string[81];
  int n;

  printf("Digite um numero: ");
  scanf("%d", &n);
  printf("Numero digitado: %d\n", n);

  setbuf(stdin, 0); /* limpa buffer da entrada padrao (stdin) */

  printf("Digite um texto: ");
  fgets(string, 81, stdin);
  printf("Texto digitado: %s", string);

  return 0;
}
```

Cópia de uma string

```
#include <stdio.h>

int main() {
  char string1[81], string2[81];
  int i = 0;

  printf("Digite uma string: ");
  fgets(string1, 81, stdin);

  /* equivalente a strcpy(string2, string1) da biblioteca string.h */
  while (string1[i]) {
    string2[i] = string1[i];
    i++;
  }
  string2[i] = string1[i];

  printf("%s", string2);

  return 0;
}
```

Cópia de uma string

```
#include <stdio.h>

int main() {
  char string1[81], string2[81];
  int i = 0;

  printf("Digite uma string: ");
  fgets(string1, 81, stdin);

  /* equivalente a strcpy(string2, string1) da biblioteca string.h */
  while (string1[i]) {
    string2[i] = string1[i];
    i++;
  }
  string2[i] = string1[i];

  printf("%s", string2);

  return 0;
}
```

Concatenação de duas strings

```
#include <stdio.h>

int main() {
  char string1[81], string2[161];
  int i = 0, j = 0;

  printf("Digite uma string: ");
  fgets(string1, 81, stdin);

  printf("Digite outra string: ");
  fgets(string2, 81, stdin);

  /* equivalente a strcat(string2, string1) da biblioteca string.h */
  while (string2[i]) i++;

  while (string1[j])
    string2[i++] = string1[j++];
  string2[i] = string1[j];

  printf("%s", string2);

  return 0;
}
```

Concatenação de duas strings

```
#include <stdio.h>

int main() {
  char string1[81], string2[161];
  int i = 0, j = 0;

  printf("Digite uma string: ");
  fgets(string1, 81, stdin);

  printf("Digite outra string: ");
  fgets(string2, 81, stdin);

  /* equivalente a strcat(string2, string1) da biblioteca string.h */
  while (string2[i]) i++;

  while (string1[j])
    string2[i++] = string1[j++];
  string2[i] = string1[j];

  printf("%s", string2);

  return 0;
}
```

Palíndromo

- Um palíndromo é uma palavra ou frase que é igual quando lida da esquerda para a direita ou da direita para a esquerda, desconsiderando-se os espaços em brancos.
- Exemplos de palíndromos: “radar”, “reviver”, “mirim”, “a sacada da casa” e “a mala nada na lama”.
- Escreva um programa que leia uma string de até 80 caracteres e teste se ela é um palíndromo.

Palíndromo

```
#include <stdio.h>

int main() {
  char string[81];
  int i = 0, j, tam = 0, palindromo = 1;

  printf("Digite uma string: ");
  fgets(string, 81, stdin);

  while (string[tam] && (string[tam] != '\n')) tam++;

  for (i = 0, j = tam - 1; palindromo && (i < j); i++, j--) {
    while ((string[i] == ' ') && (i < j)) i++; /* pula espacos a esquerda */
    while ((string[j] == ' ') && (i < j)) j--; /* pula espacos a direita */

    palindromo = (string[i] == string[j]);
  }

  if (palindromo)
    printf("Palindromo\n");

  return 0;
}
```