

Introdução

- Até agora nos concentramos em manipular números
- Manipulamos texto quando emitimos alguma mensagem
 - `printf("Digite o valor:");`
- Chamamos de **strings** estes pedaços de texto entre aspas
- Podemos colocá-las em variáveis para manipulá-las
- Questões:
 - Existe um tipo *string*?
 - Como relacionamos caracteres e strings?
 - Que operações podemos fazer com strings?

Strings e o C

- Em algumas linguagens existe o tipo *string*
 - **string** nome;
 - nome = "Joao da Silva";
 - printf("O nome eh %s", nome);
- Isso não existe em C □
- C expõe ao programador a estrutura interna da *string*
- **Aula de hoje:** entender como funcionam *strings* em C

Strings

- Strings são vetores de caracteres.



Caracteres



- Um caractere é um símbolo qualquer (letra, dígito, ...)
- Armazenamos caracteres em variáveis do tipo **char**
- Exemplos:
 - `char letra;`
 - `letra = 'a';`
 - `printf("A letra eh %c", letra);`
- Na prática caracteres são representados por números
 - Cada caractere tem um código
 - Uma variável do tipo *char* armazena esse código
 - Código ASCII: <http://pt.wikipedia.org/wiki/ASCII>

Strings e caracteres

```
char [14] nome = "Joao da Silva"
```

Strings em C



- Strings em C são vetores do tipo *char* terminados em `'\0'`
- `'\0'` é um caractere especial chamado *delimitador*
- Em constantes, o próprio compilador coloca `'\0'`.

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

```
int main(){  
    char re[8] = "lagarto";  
    printf("%s", re);  
  
    return 0;  
}
```

Na prática: "lagarto\0"

Declaração de Strings

```
char < nome-variável-string> [< tamanho> ]
```

- O tamanho da string deve prever o caractere delimitador `\0`.
- Número máximo de caracteres armazenados + 1
- Na dúvida, use um número grande:
 - `char nome[100] = "João da Silva";`

Exemplos

- Sejam duas variáveis
 - `char dia_da_semana[?]` //nomes dos dias da semana
 - `char mes[?]` //nomes dos meses do ano
- Quais seus tamanhos?
 - O dia com maior número de caracteres é segunda-feira (13)
 - O mês com maior número de caracteres é Fevereiro (9).
- Logo
 - `char dia_da_semana[14]` //13 + /0
 - `char mes[10]` //9 + /0

Inicialização de Strings

- Exemplo 1: `char primeiro_nome[15] = "Ana";`
 - O compilador insere os caracteres indicados entre as aspas duplas no vetor *primeiro_nome*, a partir da posição 0, e insere na posição 3 do vetor o caractere `\0`
- Exemplo 2: `char primeiro_nome[] = "Ana";`
 - O compilador determina o número de caracteres entre as aspas, soma um para o caractere delimitador, e cria uma string com o tamanho igual a tamanho da string + 1.

Entrada e saída de strings

- **scanf("%s", str)**

- Não colocar o & antes do nome da variável.
- Limitado a 1 palavra.
 - Encerrará a leitura do string assim que um branco for encontrado na string de entrada, ou um caractere de fim de linha ou de tabulação.

- Exemplo:

```
char nome[20];
```

```
scanf("%s", nome);
```

// Se o usuário digitar: "Joao Silva", apenas "Joao" será armazenado.

- **printf("%s", str)**

- Exemplo:

```
char nome[20];
```

```
...
```

```
printf("O nome do cliente eh: %s\n", nome)
```

Funções fgets e puts

- **gets(str)**

- Lê uma string contendo mais de uma palavra
- Exemplo:

```
char nome[20];  
gets(nome);
```

- **puts(str)**

- Permite a escrita de uma única string (constante ou variável)
- Exemplo:

```
char nome[20];  
...  
puts("O nome do cliente eh:");  
puts(nome);
```

Exemplo do uso das funções gets

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

int main(){
    char nome[30];

    puts("\nNome:");
    gets(nome);

    while (nome[0] != '\0') {
        puts("Nome informado:");
        puts(nome);

        puts("\nNome:");
        gets(nome);
    }

    return 0;
}
```

Manipulação de strings

- Utilização da biblioteca `<string.h>`
- Funções:
 - strcpy
 - strcat
 - strlen
 - strcmp

Função strcpy: copiar strings

`strcpy(< string_destino> , < string_origem >)`


- Copia o conteúdo de uma string para outra
- Strings não são um tipo
- Logo não podemos atribuir uma string a outra
- ~~string1 = string2;~~

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

int main() {
    char str[80];

    strcpy(str, "Alo");

    printf("%s", str);
    return 0;
}
```



Função strcat: concatenar strings

`strcat(<string_destino>, <string_origem>)`

- Concatenar duas strings
- A *string_origem*, sem alteração, é anexada ao final da *string_destino*.
- A *string_destino* deve ter tamanho suficiente para armazenar o resultado de *strcat*!

```
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
int main() {
    char string_origem[20],
    char string_destino[40];

    puts("Digite a primeira string: ");
    gets(string_destino);
    puts("Digite a segunda string: ");
    gets(string_origem);

    strcat(string_destino, string_origem);

    puts("\n%s\n", string_destino );
    return 0;
}
```

Função strlen: saber o tamanho de string

strlen (< string>)

-
- Retorna um valor inteiro com o número de caracteres da <string>.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
int main(){
    char re[80];
    int tam;

    puts("Digite a palavra: ");
    gets(re);

    tam = strlen(re);

    printf("Esta palavra tem %d
    caracteres.\n", tam);

    return 0;
}
```


Função strlen: saber o tamanho de string

strlen (< string>)

- Retorna um valor inteiro com o numero de caracteres da <string>.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
int main(){
    char re[80];
    int tam;

    puts("Digite a palavra: ");
    gets(re);

    tam = strlen(re);

    printf("Esta palavra tem %d
    caracteres.\n", tam);

    return 0;
}
```

Função strcmp p: compara duas strings

`strcmp p (< string_1> , < string_2>)`

- Retorna 0 se as duas strings são iguais.
- String não podem ser comparadas com o operador de comparação `=`
 - `if (string1 == string2)`

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

    puts("Digite a senha: ");
    gets(re);

    if (strcmp(re,"laranja") == 0) {
        printf ("Senha correta\n");
    } else {
        printf ("Senha invalida\n");
    }

    return 0;
}
```

Exemplo do uso das funções

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
int main(){
    char palavra[30], palavra2[30], teste[30];
    int tam, comp;

    printf("\n Informe uma string: ");
    gets(palavra);
    printf("A palavra digitada eh %s\n", palavra);
    printf("\n Informe uma segunda string: ");
    gets(palavra2);
    printf("A palavra digitada eh %s\n", palavra2);
    strcpy(teste, "aula teste");
    printf("A palavra copiada eh %s\n", teste);
    strcat(teste, " 1");
    printf("A palavra concatenada eh %s\n", teste);
    tam = strlen(palavra);
    printf("\nO tamanho da primeira string eh %d", tam);
    if (strcmp (palavra, palavra2)==0) {
        printf ("\nSao iguais: %d", comp);
    } else {
        printf ("\nSao diferentes: %d", comp);
    }

    return 0;
}
```

Outros Exemplos

- Escreva um programa que leia duas strings e as imprima na tela. Imprima também a segunda letra de cada string.

- Escreva um programa que leia uma string, conte quantos caracteres desta string são iguais a 'a' e substitua os que forem iguais a 'a' por 'b'. O programa deve imprimir o número de caracteres

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
int main(){
    char string[30];
    int tamanho, i, contaA=0;

    puts("\nInforme uma string: ");
    gets(string);

    tamanho = strlen(string);

    for (i = 0; i < tamanho; i++) {
        if (string[i] == 'a'){
            contaA++;
            string[i]='b';
        }
    }
    printf ("\nO numero de caracteres modificados eh %d", contaA);
    printf ("\nA string modificada eh %s\n", string);

    return 0;
}
```

- Faça um programa que leia o nome de 5 pessoas e mostre os nomes armazenados. Utilize vetores

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#define NUMPES 5

int main(){
    char nomes[NUMPES][10];
    int cont;

    for (i = 0; i < NUMPES; i++) {
        printf("\nInforme o nome %d: ", i);
        gets(nomes[i]);
    }

    for (i = 0; i < NUMPES; i++) {
        printf ("\nO nome armazenado na posicao %d eh %s", i, nomes[i]);
    }

    return 0;
}
```

- Uma empresa concederá um aumento de salário aos seus funcionários, variável de acordo com o cargo, conforme a tabela abaixo.

● Cargo	Percentual
----------------	-------------------

- | | |
|--------------|-----|
| ● Gerente | 10% |
| ● Engenheiro | 20% |
| ● Técnico | 30% |
- Faça um programa que leia o salário e o cargo de um funcionário e calcule o novo salário.
 - Se o cargo do funcionário não estiver na tabela, ele deverá então receber 40% de aumento.
 - Mostre o salário antigo, o novo salário e a diferença.


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

```
int main(){
    char cargo[20];
    float salario, novoSalario;

    puts("\nDigite o cargo: ");
    gets(cargo);

    puts("\nDigite o salario: ");
    scanf("%f", &salario);

    if (strcmp(cargo, "gerente") == 0){
        novoSalario = salario * 1.1;
    } else if (strcmp(cargo, "engenheiro") == 0) {
        novoSalario = salario * 1.2;
    } else if (strcmp(cargo, "tecnico") == 0) {
        novoSalario = salario * 1.3;
    } else {
        novoSalario = salario * 1.4;
    }

    printf("\nO salario antigo eh %.2f, o salario novo eh %.2f e a diferenca eh de R$ %.2f\n", salario, novoSalario, novoSalario - salario);

    return 0;
}
```