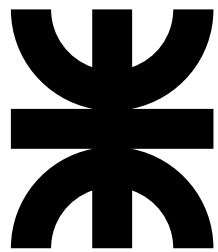
Informática II

Caracteres y cadenas

Gonzalo F. Pérez Paina



Universidad Tecnológica Nacional
Facultad Regional Córdoba
UTN-FRC

– 2018 –

Caracteres y cadenas

Arreglo de caracteres

Ejemplos:

```
char cadena1[] = {'C', 'a', 'd', '1', '\0'};
```

Caracteres y cadenas

Arreglo de caracteres

Ejemplos:

```
char cadena1[] = {'C', 'a', 'd', '1', '\0'};

char cadena2[5] = {'C', 'a', 'd', '2', '\0'};
```

Caracteres y cadenas

Arreglo de caracteres

Ejemplos:

```
char cadena1[] = {'C', 'a', 'd', '1', '\0'};

char cadena2[5] = {'C', 'a', 'd', '2', '\0'};

char cadena3[4] = {'C', 'a', 'd', '3', '\0'};
```

Caracteres y cadenas

Arreglo de caracteres

Ejemplos:

```
char cadena1[] = {'C', 'a', 'd', '1', '\0'};

char cadena2[5] = {'C', 'a', 'd', '2', '\0'};

char cadena3[4] = {'C', 'a', 'd', '3', '\0'}; // ERROR
```

Caracteres y cadenas

Arreglo de caracteres

Ejemplos:

```
char cadena1[] = {'C', 'a', 'd', '1', '\0'};

char cadena2[5] = {'C', 'a', 'd', '2', '\0'};

char cadena3[4] = {'C', 'a', 'd', '3', '\0'}; // ERROR

char cadena4[] = {67, 97, 100, 52, 0};  // ASCII
```

Caracteres y cadenas

Arreglo de caracteres

Ejemplos:

```
char cadena1[] = {'C', 'a', 'd', '1', '\0'};

char cadena2[5] = {'C', 'a', 'd', '2', '\0'};

char cadena3[4] = {'C', 'a', 'd', '3', '\0'}; // ERROR

char cadena4[] = {67, 97, 100, 52, 0};  // ASCII

char cadena5[] = "Cad5";
```

Caracteres y cadenas

Arreglo de caracteres

Ejemplos:

```
char cadena1[] = {'C', 'a', 'd', '1', '\0'};

char cadena2[5] = {'C', 'a', 'd', '2', '\0'};

char cadena3[4] = {'C', 'a', 'd', '3', '\0'}; // ERROR

char cadena4[] = {67, 97, 100, 52, 0};  // ASCII

char cadena5[] = "Cad5";

char *cadena6 = "Cad6";
```

Caracteres y cadenas

Arreglo de caracteres

Ejemplos:

```
char cadena1[] = {'C', 'a', 'd', '1', '\0'};

char cadena2[5] = {'C', 'a', 'd', '2', '\0'};

char cadena3[4] = {'C', 'a', 'd', '3', '\0'}; // ERROR

char cadena4[] = {67, 97, 100, 52, 0};  // ASCII

char cadena5[] = "Cad5";

char *cadena6 = "Cad6";
```

Imprimir

```
printf("La cadena es: %s\n", cadena1);
```

Caracteres y cadenas

Arreglo de caracteres

Ejemplos:

```
char cadena1[] = {'C', 'a', 'd', '1', '\0'};

char cadena2[5] = {'C', 'a', 'd', '2', '\0'};

char cadena3[4] = {'C', 'a', 'd', '3', '\0'}; // ERROR

char cadena4[] = {67, 97, 100, 52, 0};  // ASCII

char cadena5[] = "Cad5";

char *cadena6 = "Cad6";
```

Imprimir

```
printf("La cadena es: %s\n", cadena1);
```

- ASCII: American Standard Code for Information Interchange
- Constante de caracter: 'a', '\0', '\n', etc.

Caracteres y cadenas

Arreglo de caracteres

```c
#include <stdio.h>

int main(void)
{
  char cadena1[] = {'C', 'a', 'd', '1', '\0'};
  char cadena2[5] = {'C', 'a', 'd', '2', '\0'};
/*  char cadena3[4] = {'C', 'a', 'd', '3', '\0'};*/
  char cadena4[] = {67, 97, 100, 52, 0};
  char cadena5[] = "Cad5";
  char *cadena6 = "Cad6";

  printf("Cadena 1: %s\n", cadena1);
  printf("Cadena 2: %s\n", cadena2);
/*  printf("Cadena 3: %s\n", cadena3);*/
  printf("Cadena 4: %s\n", cadena4);
  printf("Cadena 5: %s\n", cadena5);
  printf("Cadena 6: %s\n", cadena6);

  return 0;
}
```

Caracteres y cadenas

Cadenas

Serie de caracteres tratados como una única unidad

Caracteres y cadenas

Cadenas

Serie de caracteres tratados como una única unidad

- En C, una cadena es un arreglo de caracteres terminado con el caracter nulo '\0'

Caracteres y cadenas

Cadenas

Serie de caracteres tratados como una única unidad

- En C, una cadena es un arreglo de caracteres terminado con el caracter nulo '\0'
- Puede incluir letras, dígitos y caracteres especiales como +, -, *, /, $, etc.

Caracteres y cadenas

Cadenas

Serie de caracteres tratados como una única unidad

- En C, una cadena es un arreglo de caracteres terminado con el caracter nulo '\0'
- Puede incluir letras, dígitos y caracteres especiales como +, -, *, /, $, etc.
- En C, las *literales* o *constantes* de cadenas se escriben en comillas dobles.

Caracteres y cadenas

Cadenas

Serie de caracteres tratados como una única unidad

- En C, una cadena es un arreglo de caracteres terminado con el caracter nulo '\0'
- Puede incluir letras, dígitos y caracteres especiales como +, -, *, /, $, etc.
- En C, las *literales* o *constantes* de cadenas se escriben en comillas dobles.
- Se tiene acceso mediante el puntero al primer caracter.

Caracteres y cadenas

Cadenas

Serie de caracteres tratados como una única unidad

- En C, una cadena es un arreglo de caracteres terminado con el caracter nulo '\0'
- Puede incluir letras, dígitos y caracteres especiales como +, -, *, /, $, etc.
- En C, las *literales* o *constantes* de cadenas se escriben en comillas dobles.
- Se tiene acceso mediante el puntero al primer caracter.

```
char color = "blue";
char *colorPtr = "blue";
```

- Un arreglo de caracteres debe tener el tamaño adecuado para contener la cadena mas el caracter de terminación

Caracteres y cadenas

Cadenas

Serie de caracteres tratados como una única unidad

- En C, una cadena es un arreglo de caracteres terminado con el caracter nulo '\0'
- Puede incluir letras, dígitos y caracteres especiales como +, -, *, /, $, etc.
- En C, las *literales* o *constantes* de cadenas se escriben en comillas dobles.
- Se tiene acceso mediante el puntero al primer caracter.

```
char color = "blue";
char *colorPtr = "blue";
```

- Un arreglo de caracteres debe tener el tamaño adecuado para contener la cadena mas el caracter de terminación

```
char cadena[20];
scanf("%s", cadena);
```

Caracteres y cadenas

Biblioteca para el manejo de caracteres

- Archivo de cabecera `<ctype.h>`
- Incluye funciones para pruebas y manipulación de datos tipo caracteres.

Caracteres y cadenas

Biblioteca para el manejo de caracteres

- Archivo de cabecera `<ctype.h>`
- Incluye funciones para pruebas y manipulación de datos tipo caracteres.

Algunas funcione son:

- `int isdigit(int c)`

Caracteres y cadenas

Biblioteca para el manejo de caracteres

- Archivo de cabecera `<ctype.h>`
- Incluye funciones para pruebas y manipulación de datos tipo caracteres.

Algunas funcione son:

- `int isdigit(int c)`
- `int isalpha(int c)`

Caracteres y cadenas

Biblioteca para el manejo de caracteres

- Archivo de cabecera `<ctype.h>`
- Incluye funciones para pruebas y manipulación de datos tipo caracteres.

Algunas funcione son:

- `int isdigit(int c)`
- `int isalpha(int c)`
- `int isalnum(int c)`

Caracteres y cadenas

Biblioteca para el manejo de caracteres

- Archivo de cabecera `<ctype.h>`
- Incluye funciones para pruebas y manipulación de datos tipo caracteres.

Algunas funcione son:

- `int isdigit(int c)`
- `int isalpha(int c)`
- `int isalnum(int c)`
- `int isxdigit(int c)`

Caracteres y cadenas

Biblioteca para el manejo de caracteres

- Archivo de cabecera `<ctype.h>`
- Incluye funciones para pruebas y manipulación de datos tipo caracteres.

Algunas funcione son:

- `int isdigit(int c)`
- `int isalpha(int c)`
- `int isalnum(int c)`
- `int isxdigit(int c)`
- `int islower(int c)`

Caracteres y cadenas

Biblioteca para el manejo de caracteres

- Archivo de cabecera `<ctype.h>`
- Incluye funciones para pruebas y manipulación de datos tipo caracteres.

Algunas funcione son:

- `int isdigit(int c)`
- `int isalpha(int c)`
- `int isalnum(int c)`
- `int isxdigit(int c)`
- `int islower(int c)`
- `int isupper(int c)`

Caracteres y cadenas

Biblioteca para el manejo de caracteres

- Archivo de cabecera `<ctype.h>`
- Incluye funciones para pruebas y manipulación de datos tipo caracteres.

Algunas funcione son:

- `int isdigit(int c)`
- `int isalpha(int c)`
- `int isalnum(int c)`
- `int isxdigit(int c)`
- `int islower(int c)`
- `int isupper(int c)`
- `int tolower(int c)`

Caracteres y cadenas

Biblioteca para el manejo de caracteres

- Archivo de cabecera `<ctype.h>`
- Incluye funciones para pruebas y manipulación de datos tipo caracteres.

Algunas funcione son:

- `int isdigit(int c)`
- `int isalpha(int c)`
- `int isalnum(int c)`
- `int isxdigit(int c)`
- `int islower(int c)`
- `int isupper(int c)`
- `int tolower(int c)`
- `int toupper(int c)`

Caracteres y cadenas

Funciones de conversión de cadenas

- Archivo de cabecera `<stdlib.h>` (biblioteca general de utilería).
- Convierte cadenas de dígitos a enteros y valores en punto flotante.

Caracteres y cadenas

Funciones de conversión de cadenas

- Archivo de cabecera `<stdlib.h>` (biblioteca general de utilería).
- Convierte cadenas de dígitos a enteros y valores en punto flotante.

Algunas funcione son:

- `double atof(const char *nptr)`

Caracteres y cadenas

Funciones de conversión de cadenas

- Archivo de cabecera `<stdlib.h>` (biblioteca general de utilería).
- Convierte cadenas de dígitos a enteros y valores en punto flotante.

Algunas funcione son:

- `double atof(const char *nptr)`
- `int atoi(const char *nptr)`

Caracteres y cadenas

Funciones de conversión de cadenas

- Archivo de cabecera `<stdlib.h>` (biblioteca general de utilería).
- Convierte cadenas de dígitos a enteros y valores en punto flotante.

Algunas funcione son:

- `double atof(const char *nptr)`
- `int atoi(const char *nptr)`
- `long atol(const char *nptr)`

Caracteres y cadenas

Funciones de conversión de cadenas

- Archivo de cabecera `<stdlib.h>` (biblioteca general de utilería).
- Convierte cadenas de dígitos a enteros y valores en punto flotante.

Algunas funcione son:

- `double atof(const char *nptr)`
- `int atoi(const char *nptr)`
- `long atol(const char *nptr)`
- `float strtof(const char *nptr, char **endptr)`

Caracteres y cadenas

Funciones de conversión de cadenas

- Archivo de cabecera `<stdlib.h>` (biblioteca general de utilería).
- Convierte cadenas de dígitos a enteros y valores en punto flotante.

Algunas funcione son:

- `double atof(const char *nptr)`
- `int atoi(const char *nptr)`
- `long atol(const char *nptr)`
- `float strtof(const char *nptr, char **endptr)`
- `double strtod(const char *nptr, char **endptr)`

Caracteres y cadenas

Funciones de manipulación y comparación de cadenas

- Archivo de cabecera `<string.h>`

Caracteres y cadenas

Funciones de manipulación y comparación de cadenas

- Archivo de cabecera `<string.h>`

Algunas funcione son:

- `char *strcpy(char *dest, const char *src)`

Caracteres y cadenas

Funciones de manipulación y comparación de cadenas

- Archivo de cabecera `<string.h>`

Algunas funcione son:

- `char *strcpy(char *dest, const char *src)`
- `char *strncpy(char *dest, const char *src, size_t n)`

Caracteres y cadenas

Funciones de manipulación y comparación de cadenas

- Archivo de cabecera `<string.h>`

Algunas funcione son:

- `char *strcpy(char *dest, const char *src)`
- `char *strncpy(char *dest, const char *src, size_t n)`
- `char *strcat(char *dest, const char *src)`

Caracteres y cadenas

Funciones de manipulación y comparación de cadenas

- Archivo de cabecera `<string.h>`

Algunas funcione son:

- `char *strcpy(char *dest, const char *src)`
- `char *strncpy(char *dest, const char *src, size_t n)`
- `char *strcat(char *dest, const char *src)`
- `char *strncat(char *dest, const char *src, size_t n)`

Caracteres y cadenas

Funciones de manipulación y comparación de cadenas

- Archivo de cabecera `<string.h>`

Algunas funcione son:

- `char *strcpy(char *dest, const char *src)`
- `char *strncpy(char *dest, const char *src, size_t n)`
- `char *strcat(char *dest, const char *src)`
- `char *strncat(char *dest, const char *src, size_t n)`
- `char *strcmp(const char *s1, const char *s2)`

Caracteres y cadenas

Funciones de manipulación y comparación de cadenas

- Archivo de cabecera `<string.h>`

Algunas funcione son:

- `char *strcpy(char *dest, const char *src)`
- `char *strncpy(char *dest, const char *src, size_t n)`
- `char *strcat(char *dest, const char *src)`
- `char *strncat(char *dest, const char *src, size_t n)`
- `char *strcmp(const char *s1, const char *s2)`
- `char *strncmp(const char *s1, const char *s2, size_t n)`

Caracteres y cadenas

Funciones de manipulación y comparación de cadenas

- Archivo de cabecera `<string.h>`

Algunas funcione son:

- `char *strcpy(char *dest, const char *src)`
- `char *strncpy(char *dest, const char *src, size_t n)`
- `char *strcat(char *dest, const char *src)`
- `char *strncat(char *dest, const char *src, size_t n)`
- `char *strcmp(const char *s1, const char *s2)`
- `char *strncmp(const char *s1, const char *s2, size_t n)`
- `size_t strlen(const char *s)`

Caracteres y cadenas

Arreglos de punteros – Arreglos de cadenas

- Los arreglos pueden contener punteros

Caracteres y cadenas

Arreglos de punteros – Arreglos de cadenas

- Los arreglos pueden contener punteros
- Uso común: arreglos de cadenas

Caracteres y cadenas

Arreglos de punteros – Arreglos de cadenas

- Los arreglos pueden contener punteros
- Uso común: arreglos de cadenas
  - Cada entrada del arreglo es una cadena

Caracteres y cadenas

Arreglos de punteros – Arreglos de cadenas

- Los arreglos pueden contener punteros
- Uso común: arreglos de cadenas
  - Cada entrada del arreglo es una cadena
  - Cada entrada es un puntero al primer caracter de la cadena

Caracteres y cadenas

Arreglos de punteros – Arreglos de cadenas

- Los arreglos pueden contener punteros
- Uso común: arreglos de cadenas
  - Cada entrada del arreglo es una cadena
  - Cada entrada es un puntero al primer caracter de la cadena

Ejemplo

```
char *suit[4] = {"Heart", "Diamonds", "Clubs", "Spades"};
```

Caracteres y cadenas

Arreglos de punteros – Arreglos de cadenas

- Los arreglos pueden contener punteros
- Uso común: arreglos de cadenas
  - Cada entrada del arreglo es una cadena
  - Cada entrada es un puntero al primer caracter de la cadena

Ejemplo

```
char *suit[4] = {"Heart", "Diamonds", "Clubs", "Spades"};
```

¿Qué pasa si se almacenan las cadenas en un arreglo doble?

Caracteres y cadenas

Arreglos de punteros – Arreglos de cadenas

- Los arreglos pueden contener punteros
- Uso común: arreglos de cadenas
  - Cada entrada del arreglo es una cadena
  - Cada entrada es un puntero al primer caracter de la cadena

Ejemplo

```
char *suit[4] = {"Heart", "Diamonds", "Clubs", "Spades"};
```

¿Qué pasa si se almacenan las cadenas en un arreglo doble?

Ver prototipo de la función `main`.

Caracteres y cadenas

Arreglos de punteros – Arreglos de cadenas

- Los arreglos pueden contener punteros
- Uso común: arreglos de cadenas
  - Cada entrada del arreglo es una cadena
  - Cada entrada es un puntero al primer caracter de la cadena

Ejemplo

```
char *suit[4] = {"Heart", "Diamonds", "Clubs", "Spades"};
```

¿Qué pasa si se almacenan las cadenas en un arreglo doble?

Ver prototipo de la función `main`.

Escribir un programa que:

1. Defina un *arreglo de cadenas* y las imprima en un buche `for`

Caracteres y cadenas

Arreglos de punteros – Arreglos de cadenas

- Los arreglos pueden contener punteros
- Uso común: arreglos de cadenas
  - Cada entrada del arreglo es una cadena
  - Cada entrada es un puntero al primer caracter de la cadena

Ejemplo

```
char *suit[4] = {"Heart", "Diamonds", "Clubs", "Spades"};
```

¿Qué pasa si se almacenan las cadenas en un arreglo doble?

Ver prototipo de la función `main`.

Escribir un programa que:

1. Defina un *arreglo de cadenas* y las imprima en un buche `for`
2. Imprima en un bucle `for` las cadenas pasadas a la función `main`