

# FUNDAMENTOS DE PROGRAMACIÓN

## UNIDAD 3: STRING



Facultad de Ingeniería  
Universidad Nacional de Jujuy

# STRING C++ - DEFINICIÓN

- CADENA DE CARACTERES: UNA CADENA DE CARACTERES ES UN CONJUNTO DE CARACTERES (INCLUIDO EL ESPACIO EN BLANCO) RECONOCIDOS POR LA COMPUTADORA, LOS QUE SE ALMACENAN EN POSICIONES DE MEMORIAS CONTIGUAS.
- LA CADENA DE CARACTERES ES UNA CONCATENACION DE CARACTERES QUE SE TRATAN COMO UN ÚNICO ELEMENTO.
- LAS CADENAS DE CARACTERES PERMITEN ALMACENAR: PALABRAS, FRASES, CUALQUIER TIPO DE CARÁCTER COMO ESPACIOS EN BLANCO, SIGNOS, ETC.
- C++ DISPONE DE UNA LIBRERÍA <STRING> QUE INCLUYE EL TIPO STRING, USADO PARA TRABAJAR CON CADENA DE CARACTERES.
- SE PUEDE TRABAJAR CON CADENAS DE CARACTERES TIPO C (ARRAY DE CHAR).
- LA CADENA QUE NO CONTIENE NINGÚN CARÁCTER SE DENOMINA VACÍA O NULA.

F	u	n	d	a	m	e	n	t	o	s		d	e		p	r	o	g	r	a	m	a	c	i	o	n
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26

# STRING C++ - OPERACIONES (1)

## ► LIBRERÍA

```
#include <string>;
```

## ► DEFINICIÓN DE VARIABLES

```
string frase="esta es mi cadena de caracteres"; //definición e inicialización  
char car='a'; //no es una cadena es un carácter
```

## ► SALIDA

```
cout<< "muestra esta cadena por pantalla";  
cout<<frase; // muestra por pantalla esta es mi cadena de caracteres
```

# STRING C++ - OPERACIONES (2)

## ➡ ENTRADA

`cin >> frase; //almacena palabras o caracteres hasta que encuentra un espacio en blanco`

`getline(cin,frase); //almacena cadenas con espacios en blanco hasta encontrar un salto o fin de línea o carácter delimitador`

`getline(cin,frase,'t'); //almacena cadenas con espacios en blanco hasta encontrar un carácter delimitador`

`cin.ignore(); //limpia el buffer`

# STRING C++ - OPERACIONES (3)

## ASIGNACIÓN

palabras=maspalabras; //palabras, maspalabras, variables de tipo string, deben ser del mismo tamaño.

## COMPARACIÓN

if (palabras==maspalabras)

if (palabras>maspalabras)

if (palabras<maspalabras)

if (palabras!=maspalabras)

TABLA DE CARACTERES DEL CÓDIGO ASCII

1	25	49	73	97	121	145	169	193	217	241
2	26	50	74	98	122	146	170	194	218	242
3	27	51	75	99	123	147	171	195	219	243
4	28	52	76	100	124	148	172	196	220	244
5	29	53	77	101	125	149	173	197	221	245
6	30	54	78	102	126	150	174	198	222	246
7	31	55	79	103	127	151	175	199	223	247
8	32	56	80	104	128	152	176	200	224	248
9	33	57	81	105	129	153	177	201	225	249
10	34	58	82	106	130	154	178	202	226	250
11	35	59	83	107	131	155	179	203	227	251
12	36	60	84	108	132	156	180	204	228	252
13	37	61	85	109	133	157	181	205	229	253
14	38	62	86	110	134	158	182	206	230	254
15	39	63	87	111	135	159	183	207	231	255
16	40	64	88	112	136	160	184	208	232	256
17	41	65	89	113	137	161	185	209	233	257
18	42	66	90	114	138	162	186	210	234	258
19	43	67	91	115	139	163	187	211	235	259
20	44	68	92	116	140	164	188	212	236	260
21	45	69	93	117	141	165	189	213	237	261
22	46	70	94	118	142	166	190	214	238	262
23	47	71	95	119	143	167	191	215	239	263
24	48	72	96	120	144	168	192	216	240	264



# STRING C++ - OPERACIONES (4)

## ▶ LONGITUD

```
int longitud=palabras.size(); //devuelve la longitud de la cadena  
int longitud=palabras.lenght();
```

## ▶ VACIA

```
vacía=palabras.empty(); //indica si palabras está vacía
```

## ▶ CONCATENACIÓN

```
maspalabras+=palabras; //concatena o une cadenas, añade al  
final palabras a maspalabras
```

```
maspalabras=maspalabras+ "hola."; //concatena o une cadenas,  
añade al final hola. al contenido de maspalabras
```

```
maspalabras.append(palabras); //concatena, añade al final  
palabras al contenido de maspalabras
```

# STRING C++ - OPERACIONES (5)

## ➤ INSERCIÓN

`palabras.insert (pos,palabra);` //inserta palabra en la posición pos

## ➤ BORRADO

`palabras.erase (pos,num);` //borra num caracteres de palabras desde la posición pos

## ➤ REDIMENSIONAMIENTO

`palabras.resize(10);` //cambia el tamaño de una string  
`palabras.clear();`

## ➤ SUSTITUCIÓN

`palabras.replace (pos,num,palabra);` //reemplaza por palabra, num caracteres de palabras comenzando en la posición pos

## ➤ BÚSQUEDA

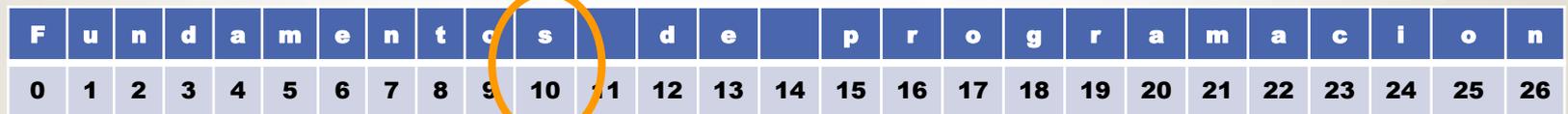
`i=palabras.find (palabra, pos);` //busca palabra como una subcadena dentro de palabras desde la posición pos, devuelve la posición donde la encuentra.

# STRING C++ - OPERACIONES (6)

- ➔ ACCESO. Para acceder a una posición específica de una string debe indicarse el nombre de la string y el índice del elemento requerido.

```
char car=palabras.at(10); //accede a s
```

```
char car=palabras[10];
```



A diagram showing a string array with 27 indices. The characters are: F, u, n, d, a, m, e, n, t, o, s, d, e, p, r, o, g, r, a, m, a, c, i, o, n. The character 's' at index 10 is circled in orange, with an arrow pointing to it from the code above.

F	u	n	d	a	m	e	n	t	o	s	d	e	p	r	o	g	r	a	m	a	c	i	o	n		
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26

- ➔ SUBCADENAS

```
string subcad=palabras.substr(0,11); //devuelve la subcadena  
formada por 11 caracteres desde la posición 0.
```



A diagram showing a string array with 27 indices. The characters are: F, u, n, d, a, m, e, n, t, o, s, d, e, p, r, o, g, r, a, m, a, c, i, o, n. The first 11 characters (indices 0 to 10) are circled in orange, with an arrow pointing to the circle from the code above.

F	u	n	d	a	m	e	n	t	o	s	d	e	p	r	o	g	r	a	m	a	c	i	o	n		
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26

# STRING C++ - RETOS

1. REALIZA UN CODIGO DONDE PIDAS 5 NOMBRES Y 5 EDADES Y LUEGO MUESTRA LOS RESULTADOS POR PANTALLA
2. REALIZA UN EJERCICIO QUE CUENTE EL NRO. DE PALABRAS DE UNA CADENA DE CARACTERES
3. COMPROBAR SI UNA PALABRA ES PALÍNDROMO (AMA – MENEM – NEUQUEN)
4. ENCONTRAR LA PALABRA DE MAYOR LONGITUD DE UNA FRASE
5. AÑADIR EL PREFIJO INTERNACIONAL A UN TELÉFONO
6. CONTAR LA FRECUENCIA CON LAS QUE APARECE UNA VOCAL EN UNA FRASE
7. BUSCAR UNA PALABRA EN UNA FRASE