Part III  Strings  (Summary)

❖ Definition

A one-dimensional array that stores a collection of data items of type “char”

\[\text{char}\ string\_name[\text{string\_size}]\]

Example:  \(\text{char str[6];}\)

NOTE:
An extra memory location is added at the end of the string. It will be initialised with a special character “NULL”: ‘\0’ when using string-processing functions

❖ Using Strings

\(\text{string\_name[index]}\)

Example:  \(\text{str[0]}\)

\(\text{str[0]} = \text{‘X’};\)  \(=>\) change the first character of the string
String Initialisation

- at declaration:
  - char-by-char:
    ```cpp
    char name[5] = {'J', 'o', 'h', 'n', '\0'};
    ```
  - as a string:
    ```cpp
    char name[5] = "John";
    ```

- during execution

  **user input => char-by-char**
  ```cpp
  char name[5], ch;
  for (index = 0; index < 5; index++)
    scanf("%c%c", &name[index], &ch);
  name[index] = '\0';
  ```

  **user input => as a string**
  ```cpp
  char name[5];
  scanf("%s", name);
  ```

  **without user input (automatic) => char-by-char**
  ```cpp
  char name[5];
  for (index = 0; index < 5; index++)
    name[index] = 'a' + index;
  name[index] = '\0';
  ```

  **without user input (automatic) => as a string**
  ```cpp
  char name[5];
  strcpy(name, "John");
  ```
Input / Output String Functions

Input Functions

```c
#include <stdio.h>
char name[20];
scanf("%s", name);
gets(name);
```

Output Functions

```c
#include <stdio.h>
char name[20];
printf("%s", name);
puts(name);
```

string.h Library

- Contains a set of functions for processing strings
- string.h library must be included in the program
  ```c
  #include <string.h>
  ```
  - Functions for two strings comparison
    ```c
    int strcmp(char* string1, char* string2);
    int strncmp(char* string1, char* string2, int n);
    ```

Example:

```c
char string[5] = "John"
char name[5] = "Test"
int val;
val = strcmp (string, name);  val < 0
```

ASCII Code = 74

ASCII Code = 84
• Functions for **copying** a string into another string

```c
char* strcpy(char* dest, const char* src);
char* strncpy(char* dest, char* src, int n);
```

Example:

```c
char string[5] = "John"
char name[5] = "Test"
strcpy (name, string);
```

<table>
<thead>
<tr>
<th>string</th>
<th>name</th>
</tr>
</thead>
<tbody>
<tr>
<td>‘J’</td>
<td>‘o’</td>
</tr>
<tr>
<td>‘o’</td>
<td>‘h’</td>
</tr>
<tr>
<td>‘h’</td>
<td>‘n’</td>
</tr>
<tr>
<td>‘n’</td>
<td>‘/0’</td>
</tr>
</tbody>
</table>

• Functions for **appending** a string to another string

```c
char* strcat(char* dest, const char* src);
int strncat(char* dest, char* src, int n);
```

Example:

```c
char string[5] = "John"
char name[10] = "Sean"
strncat (name, string,1);
```

<table>
<thead>
<tr>
<th>string</th>
<th>name</th>
</tr>
</thead>
<tbody>
<tr>
<td>‘J’</td>
<td>‘o’</td>
</tr>
<tr>
<td>‘o’</td>
<td>‘h’</td>
</tr>
<tr>
<td>‘h’</td>
<td>‘n’</td>
</tr>
<tr>
<td>‘n’</td>
<td>‘/0’</td>
</tr>
<tr>
<td>‘S’</td>
<td>‘e’</td>
</tr>
<tr>
<td>‘e’</td>
<td>‘a’</td>
</tr>
<tr>
<td>‘a’</td>
<td>‘n’</td>
</tr>
<tr>
<td>‘n’</td>
<td>‘J’</td>
</tr>
<tr>
<td>‘J’</td>
<td>‘/0’</td>
</tr>
</tbody>
</table>
- **Function for determining the length of a string:**
  ```c
  int strlen(char* string);
  ```

  **Example:**
  ```c
  char name[10] = "Sean";
  int val;
  val = strlen(name);  =>  val = 4;
  ```

- **ctype.h Library**
  - Contains several functions useful for testing and mapping characters
  - `ctype.h` library must be included in the program
    ```c
    #include <ctype.h>
    ```
  - **Character testing functions**
    ```c
    int isdigit (int ch);
    int islower (int ch);
    int isspace (int ch);
    ```
  - **Character case mapping functions**
    ```c
    int tolower (int ch);
    int toupper (int ch);
    ```
Strings passed as parameters to a function

- "by-address" strategy is used
- '\0' indicates the end of the string => no need to indicate explicitly the length of the string via another parameter

Example:

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

void print_name(char str[])
{
    int i;

    printf("The name is spelled: ");
    for (i = 0; i < strlen(str); i++)
        printf("%c ", str[i]);
}

int main ()
{
    char text[100];
    printf("Enter a name: ");
    gets(text);
    print_name(text);
    return(EXIT_SUCCESS);
}
```