ARRAYS, C-STRINGS AND POINTERS
DYNAMIC ALLOCATION OF MEMORY
Examples

==========

```c
// Arrays & C-Strings
// ex_00.c
// A program that reads the evolution of a person's weight
// during some months of a year
// JAS

#include <stdio.h>

#define MAX_NAME_LENGTH 50 // avoid "magic numbers"
#define NUM_MONTHS 12

int main()
{
  char name[MAX_NAME_LENGTH];
  int weight[NUM_MONTHS];
  int i;

  printf("Name ? ");
  scanf("%s", name);  // try with "Lou Costello" :-)

  for (i = 0; i < NUM_MONTHS; i++)
  {
    printf("weight[%d] ? ", i + 1);
    scanf("%d", &weight[i]);
  }

  for (i = 0; i < NUM_MONTHS; i++)
  {
    printf("weight[%d] = %d\n", i + 1, weight[i]);
  }

  // TO BE DONE ...??? complete the data processing

  return 0;
}
```

```c
// ============================================================================
// Arrays & C-Strings
// ex_01.c
// A program that reads the evolution of a person's weight
// during some months of a year
// JAS

#include <stdio.h>

#define MAX_NAME_LENGTH 50 // avoid "magic numbers"
#define NUM_MONTHS 3

int main()
{
  char name[MAX_NAME_LENGTH]; // person's name
  int weight[NUM_MONTHS];     // weight evolution along months
  int i;

  // Read the name and weights
  printf("Name ? ");
  scanf("%s", name); // try with "Lou Costello" :-)

  for (i = 0; i < NUM_MONTHS; i++)
  {
    printf("weight[%d] ? ", i + 1);
    scanf("%d", &weight[i]);
  }

  // Show the weights
  printf("\n");
  for (i = 0; i < NUM_MONTHS; i++)
  {
    printf("weight[%d] = %d\n", i + 1, weight[i]);
  }

  // TO BE DONE ...??? complete the data processing

  return 0;
}
```

```c
// ============================================================================
// Arrays & C-Strings
// ex_02.c
// A program that reads the evolution of a person's weight
// during the 12 months of a year
// Using fgets() to read names with more than one word
// JAS

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

#define MAX_NAME_LENGTH 50 // avoid "magic numbers"
#define NUM_MONTHS 3

int main()
{
  char name[MAX_NAME_LENGTH]; // person's name
  int weight[NUM_MONTHS];     // weight evolution along months
  int i;

  // Read the name and weights
  printf("Name ? "); // try with "Lou Costello" :-)
  fgets(name, sizeof(name), stdin);
  // NOTE:
  // fgets() is safer than gets()
  // fgets() always inserts a '\0' at the end of the string
  // while gets() may not insert.
  // Newline is read only if there is available space in 'name'

  // TRY WITH 'name' =
  // 1) 123
  // 2) 1234567890
  // 3) 123456789
  // AND INTERPRET THE RESULTS
  // (VERIFY THAT IN SOME CASES THE NEWLINE APPEARS IN 'name')

  // SE INTRODUZIR MAIS DO QUE 9 CARACTERES
  // O CICLO SEGUINTE NÃO FUNCIONA COMO DEVIA.
  // OS CARACTERES FICAM NO BUFFER

  if ((strlen(name) == (sizeof(name) - 1)) && name[strlen(name) - 1] != '\n')
    while (getchar() != '\n'); // to clean the buffer

  for (i = 0; i < NUM_MONTHS; i++)
  {
    printf("weight[%d] ? ", i + 1);
    scanf("%d", &weight[i]);
    perror("main"); // try with an invalid input; error detected?!
    // scanf() has a return value ... look for its meaning
    /*
    {
    fprintf(stderr,"Invalid input !\n");
    system("pause");
    exit(1); // try with an invalid input; error detected?!
    }
    */
  }
```

```
  // Show the weights
  printf("\n");
  printf("%s", name);
  for (i = 0; i < NUM_MONTHS; i++)
  {
    printf("weight[%d] = %d\n", i + 1, weight[i]);
  }

  return 0;
}
```

```c
// ========================================================================
// Arrays & C-Strings
// ex_03.c
// JAS

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

#define MAX_NAME_LENGTH 10  // TRY WITH OTHER VALUES

int main()
{
  // Strings declarations
  char name1[MAX_NAME_LENGTH] = "Ann Sousa"; // try with a longer name
  char name2[MAX_NAME_LENGTH];
  char *name3; // assigned to a constant (see below)
  char *name4;
  char *name5 = NULL;

  printf("name1=%s\n", name1);
  //name2="John"; // --> COMPILATION ERROR

  name3 = "John Dalton";
  printf("name3=%s\n", name3);

  name4 = (char *)malloc(MAX_NAME_LENGTH*sizeof(char));
  printf("Nome4 ? "); fgets(name4, MAX_NAME_LENGTH, stdin);
  printf("name4=%s\n", name4);

  // WHICH IS THE DIFFERENCE AMONG THE 3 SEQUENCES OF STATEMENTS BELOW ?

  //1) -
  // name5=name4;
  // printf("%s\n",name5);

  //2) - THE PROGRAM MAY CRASH ... WHY?
  // strcpy(name5,name4); // SYNTAX: strcpy(destination,source)
  // printf("%s\n",name5);

  //3) - CORRECTION OF THE ERROR FROM 2) ... WHY?
  name5 = (char *)malloc((strlen(name4) + 1)*sizeof(char));
  strcpy(name5, name4); // SYNTAX: strcpy(destination,source)
  printf("name5=%s\n", name5);

  //4)
  name5 = &name1[4];
  printf("name5(new)=%s\n", name5);

  return 0;
}
```

```c
// ========================================================================
// Relationship between Arrays and Pointers
// ex_04.c
// JAS

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

#define MAX_LEN 10

int main()
{
  int i;
  int a[MAX_LEN];
  int *b;

  for (i = 0; i < MAX_LEN; i++)
    a[i] = i;

  printf("a[] = \n");
  for (i = 0; i < MAX_LEN; i++)
    printf("a[%d]=%d\n", i, *(a + i));

  b = a;

  printf("b[] = a[] = \n");
  for (i = 0; i < MAX_LEN; i++)
    printf("a[%d]=%d\n", i, b[i]);

  return 0;
}
```

```c
// ========================================================================
// Arrays of strings
// ex_05a.c
// JAS

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

#define MAX_NAME_LENGTH 20
#define MAX_NAMES 3

void show(char nms[][MAX_NAME_LENGTH], int n)
{
  int i;
  for (i = 0; i < n; i++)
    printf("%s\n", nms[i]);
}

int main()
{
  char names[MAX_NAMES][MAX_NAME_LENGTH];

  strcpy(names[0], "Ann");  // How many chars were allocated for this name?
  strcpy(names[1], "John Dalton");
  show(names, 2);

  return 0;
}

// ========================================================================
// Arrays of strings
// ex_05b.c
// JAS

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

#define MAX_NAME_LENGTH 20
#define MAX_NAMES 3

void show(char *nms[], int n) // NOTE THE DIFFERENCE FROM PREVIOUS EXAMPLE
{                             // EXPLAIN THE COMPILATION WARNING
  int i;
  for (i = 0; i < n; i++)
    printf("%s\n", nms[i]);
}

int main()
{
  char names[MAX_NAMES][MAX_NAME_LENGTH];

  strcpy(names[0], "Ann");
  strcpy(names[1], "John Dalton");
  show(names, 2);

  return 0;
}
```

```c
// ========================================================================
// Arrays of strings
// ex_06a.c
// JAS

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

#define MAX_NAME_LENGTH 20
#define MAX_NAMES 3 // >=2

void show(char *nms[], int n)
// void show(char **nms, int n)  // ALTERNATIVE
{
  int i;
  for (i = 0; i < n; i++)
    printf("%s\n", nms[i]);
}

int main()
{
  char *names[MAX_NAMES];

  int i;
  for (i = 0; i < MAX_NAMES; i++)
    names[i] = (char *)malloc(MAX_NAME_LENGTH*sizeof(char));

  strcpy(names[0], "Ann");
  strcpy(names[1], "John Dalton");
  show(names, 2);

  for (i = 0; i < MAX_NAMES; i++)
    free(names[i]);

  return 0;
}
```

```c
// ========================================================================
// Arrays of strings
// ex_06b.c
// JAS

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

#define MAX_NAME_LENGTH 10

void readNames(char **nms, int n)
{
  int i;
  for (i = 0; i < n; i++)
  {
    printf("Name [%d] ? ", i + 1);
    fgets(nms[i], MAX_NAME_LENGTH, stdin);
  }
}

void showNames(char **nms, int n)
{
  int i;
  for (i = 0; i < n; i++)
    printf("%s", nms[i]);
}

int main()
{
  char **names;
  int i, n;

  printf("How many names ? ");
  scanf("%d", &n);
  while (getchar() != '\n'); //fflush(stdin);

  // Dinamically allocate memory
  names = (char **)malloc(n*sizeof(char *));
  for (i = 0; i < n; i++)
    names[i] = (char *)malloc(MAX_NAME_LENGTH*sizeof(char));
  // The space allocated for each name is fixed ... :-(
  // TO DO BY STUDENTS: allocate only the necessary space for each name

  // Read the names
  readNames(names, n);

  // Show the names
  showNames(names, n);

  // Free the dinamically allocated memory
  for (i = 0; i < n; i++)
    free(names[i]);
  free(names);

  return 0;
}
```

```c
// ========================================================================
// Functions and Pointers
// ex_07.c
// JAS

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

#define MAX_LEN 10

void test1(void)
{
  printf("TEST1\n");
  printf("Hello\n");
}

void test2(int n)
{
  int i;
  printf("TEST2\n");
  for (i = 0; i < n; i++)
    printf("Hello no. %d\n", i);
}

void test3(int n, void(*f) (void))
{
  int i;
  printf("TEST3\n");
  for (i = 0; i < n; i++)
    f();
}

int main()
{
  void(*func1) (void);
  void(*func2) (int);

  func1 = test1;
  func1();

  func2 = test2;
  func2(3);

  test3(2, test1);

  return 0;
}
```