## Variabili, espressioni, I/O (pag. 96)

(8')

- Questa è una memoria RAM composta da 16 **parole** o **locazioni** (= gruppo di celle che va letto o scritto tutto insieme) ognuna composta da 8 bit (1 Byte)

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| D0 (B0000) [0x0] | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |
| D1 (B0001) [0x1] | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| D2 (B0010) [0x2] | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| D3 (B0011) [0x3] | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 |
| D4 (B0100) [0x4] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| D5 (B0101) [0x5] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| D6 (B0110) [0x6] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| D7 (B0111) [0x7] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| D8 (B1000) [0x8] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| D9 (B1001) [0x9] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| D10 (B1010) [0xA] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| D11 (B1011) [0xB] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| D12 (B1100) [0xC] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| D13 (B1101) [0xD] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| D14 (B1110) [0xE] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| D15 (B1111) [0xF] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Cella di memoria (1 bit di informazione)

Indirizzo locazione in esadecimale
Indirizzo locazione in binario
Indirizzo locazione in decimale

- Se ho locazioni da 8 bit e il mio numero numero è lungo 16 bit, in memoria esso occuperà 2 locazioni

Numeri riconosciuti dal C:
**123** Intero a 8 bit (essendo < 255);
**2745** Intero a 16 bit;
**3,14** Reale;
**2,4e-6** Reale notazione esponenziale;
**0x7D** Intero in esadecimale

**\x** num esadecimale inserisce carattere ASCII codificato con quel esadecimale

**'a'** x dare la **lettera 'a'** piccola la racchiudo tra apici
**"abcde"** inserisce una **stringa (sequenza di caratteri)**
**\x61** inserisce la **'a'** perché l'esadec 61 codifica 'a' piccola

**Caratteri che sono comandi** Il C riconosce queste **sequenze di escape**

| | |
|---|---|
| \n | newline (a capo); |
| \f | form feed (pagina seguente) |
| \r | carriage return (ritorno carrello); |
| \t | tab (tabulazione); |
| \b | backspace (indietro di uno spazio) |
| \\ | backslash (inserisce una barra rovesciata) |

```
4    int A=0X7D;          // senza includere iostream dà errore su printf e system
5    printf("%d \n", A); // programma principale
6    system("pause");
7  }
```

```
C:\Users\UTENTE\Documents\
125
```

Anche /* … */ è un commento, che può estendersi su più linee.
Ciò che si scrive dopo // è un **commento** e viene ignorato dal compilatore.

```
// Dichiaro A variabile intera e ci metto l'esadecimale 7D (7x16+13=125)
// Stampo dove sta il %d la variabile intera A poi stampo vai a nuova linea
// Fermo l'esecuzione in modo da poterlo leggere lo schermo
```

```
[Error] 'A' was not declared in this scope
[Error] 'printf' was not declared in this scope
[Error] 'system' was not declared in this scope
```

**Parole chiavi (*keyword*) del C**

es1a_pag109.cpp | prova_esadecimali_2.cpp | prova_e

```
1  #include <iostream>     // senza inc
2  int main()              // programma
3  {
```

```
inserisci il primo numero...1
la somma parziale e': 1
inserisci il secondo numero...2
la somma parziale e': 3
inserisci il terzo numero...3
la somma parziale e': 6
```

Tipo
**char** (1 Byte) carattere o intero corto
**int** (2 Byte) intero fino a $2^{15}$-1 se **signed** e fino a $2^{16}$-1 se **unsigned** (solo positivi)
**long int** (4 Byte)
**float** (4 Byte) flottante = numero con la virgola mobile
**double** (8 Byte) numero con la virgola in doppia precisione
**long double** (10 Byte)

**%c** segnaposto x un char (carattere)
**%d** segnaposto x intero decimale
**%f** segnaposto x un float
**%s** segnaposto x una stringa

```
4    char X='\x61';      // Dichiara X variabile char e ci mette l'ASCII di a in esadecimale
5    printf("%c \n", X); // Stampa X sta il segnaposto (%c = dato di tipo char)
6    system("pause");
7  }
```

```
a
Premere un tasto per continuare . . .
```

```
1  #include <iostream>
2  int main()
3  {
4      int NUM, SOMMA=0;
5      printf("inserisci il primo numero...");
6      scanf("%d", &NUM);
7      SOMMA=SOMMA+NUM;
8      printf("la somma parziale e': %d\n", SOMMA);
9      printf("inserisci il secondo numero...");
10     scanf("%d", &NUM);
11     SOMMA=SOMMA+NUM;
12     printf("la somma parziale e': %d\n", SOMMA);
13     printf("inserisci il terzo numero...");
14     scanf("%d", &NUM);
15     SOMMA=SOMMA+NUM;
16     printf("la somma parziale e': %d\n", SOMMA);
17     system("pause");
18 }
```