

Somma di numeri binari, Trasposta di una matrice, File

Dott. Emanuele Pio Barracchia

Somma di numeri binari

- Scrivere un programma in linguaggio C, che dati due numeri binari sotto forma di array, ne calcola la somma

1	1	0	0
---	---	---	---

+

1	0	1	0
---	---	---	---

=

1	0	1	1	0
---	---	---	---	---



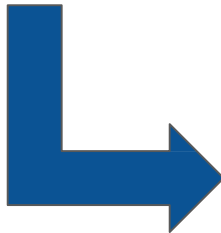
Somma di numeri binari

```
int main(){
    int num1[] = {0, 0, 1, 1};
    int num2[] = {0, 1, 0, 1};

    int size1 = sizeof(num1)/sizeof(num1[0]);
    int size2 = sizeof(num2)/sizeof(num2[0]);

    //Stampo a video i due numeri binari
    printf("Primo numero binario: ");
    for (int i = size1-1; i >= 0; i--){
        printf("%d", num1[i]);
    }
    printf("\n");

    printf("Secondo numero binario: ");
    for (int i = size2-1; i >= 0; i--){
        printf("%d", num2[i]);
    }
    printf("\n");
```



```
    int maxSize;
    if (size1>size2)
        maxSize = size1;
    else
        maxSize = size2;

    int result[maxSize+1];
    int carry = 0;

    for (int i = 0; i < maxSize; i++){
        int somma = num1[i] + num2[i] + carry;

        if (somma > 1){
            somma = somma - 2;
            carry = 1;
        } else {
            carry = 0;
        }

        result[i] = somma;
    }
    result[maxSize] = carry;

    //Stampo a video il risultato
    printf("La somma è pari a : ");
    for (int i = maxSize; i >= 0; i--){
        printf("%d", result[i]);
    }
    printf("\n");

    return 0;
```

```
}
```



Matrici

- Una matrice è un array bidimensionale
- Es. Definire una matrice di numeri interi in linguaggio C composta di N righe e M colonne

```
int matrix[n][m];
```



Matrici - Esercizi

- Scrivere un programma in linguaggio C che, data una matrice, genera la sua trasposta
 - NB. La matrice trasposta è la matrice ottenuta scambiando le righe con le colonne

$$\begin{bmatrix} 9 & 0 & 1 \\ 0 & 11 & 1 \\ 1 & 1 & 4 \\ 1 & 0 & 1 \end{bmatrix} \xrightarrow{\text{Transpose}} \begin{bmatrix} 9 & 0 & 1 & 1 \\ 0 & 11 & 1 & 0 \\ 1 & 1 & 4 & 1 \end{bmatrix}$$



Matrici - Esercizi

```
int main(){
    int matrix[4][3] = {
        {9,0,1},
        {0,11,1},
        {1,1,4},
        {1,0,1}
    };

    int numRows = sizeof(matrix)/sizeof(matrix[0]);
    int numCol = (sizeof(matrix[0])/sizeof(matrix[0][0]));

    //stampo a video la matrice
    printf("Matrice di partenza: \n");

    for (int i = 0; i < numRows; i++){
        for (int j = 0; j < numCol; j++){
            printf("%d ", matrix[i][j]);
        }
        printf("\n");
    }
}
```

```
printf("-----\n");

//genero la traposta della matrice
int transMatrix[numCol][numRows];
for (int i = 0; i < numRows; i++){
    for (int j = 0; j < numCol; j++){
        transMatrix[j][i] = matrix[i][j];
    }
}

printf("Matrice trasposta: \n");

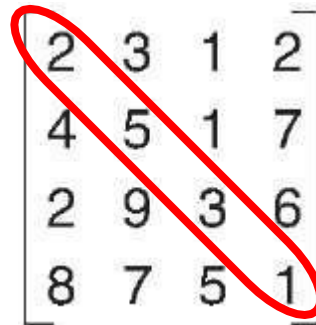
for (int i = 0; i < numCol; i++){
    for (int j = 0; j < numRows; j++){
        printf("%d ", transMatrix[i][j]);
    }
    printf("\n");
}

return 0;
}
```



Matrici - Esercizi

- Scrivere un programma in linguaggio C che, data una matrice quadrata, somma gli elementi presenti sulla diagonale



2	3	1	2
4	5	1	7
2	9	3	6
8	7	5	1



Matrici - Esercizi

```
int main(){
    int matrix[4][4] = {
        {2,3,1,2},
        {4,5,1,7},
        {2,9,3,6},
        {8,7,5,1}
    };

    int numRows = sizeof(matrix)/sizeof(matrix[0]);
    int numCol = (sizeof(matrix[0])/sizeof(matrix[0][0]));

    //stampo a video la matrice
    printf("Matrice di partenza: \n");

    for (int i = 0; i < numRows; i++){
        for (int j = 0; j < numCol; j++){
            printf("%d ", matrix[i][j]);
        }
        printf("\n");
    }

    printf("-----\n");

    int sum = 0;

    for (int i = 0; i < numRows; i++){
        sum = sum + matrix[i][i];
    }

    printf("La somma degli elementi sulla diagonale è pari a: %d \n", sum);
    return 0;
}
```



File

- In C le operazioni di input/output avvengono mediante l'uso di **stream**, ovvero astrazioni rappresentative di un file o di un dispositivo fisico che sono manipolabili mediante l'uso di puntatori
- Operazioni frequenti su stream sono: **apertura, accesso e chiusura**
- Esistono i buffer per gli stream in modo tale da evitare ritardi o interruzioni nella fase di lettura e scrittura.
 - NB: il contenuto di un buffer non viene mandato al file o al dispositivo finchè il buffer non è svuotato o chiuso



Stream

Gli stream predefiniti nel linguaggio C sono:

- **stdin**, per i flussi in input. Il dispositivo di default è la tastiera
- **stdout**, per i flussi in output. Il dispositivo di default è la console
- **stderr**, per i messaggi di errore. Il dispositivo di default è la console



Apertura di un file

La funzione utilizzata per aprire un file è la seguente:

FILE *fopen(char *nome_file, char *modalità_di_apertura)

La funzione **fopen** accetta le seguenti modalità di apertura di un file:

Parametro	Azione	Descrizione
r	(read) lettura	Lettura da un file esistente
w	(write) scrittura	Scrive un nuovo file o sovrascrive un file esistente
a	(append) aggiungere dati	Aggiunge dati partendo dalla fine del file



Accesso al file

Una volta aperto un file, è possibile accedervi mediante due funzioni:

- **int** fprintf(**FILE** *stream, **char** *formato, argomenti ...)
- **int** fscanf(**FILE** *stream, **char** *formato, argomenti ...)



Chiusura di un file

Infine, gli stream, una volta utilizzati, devono essere prima “puliti” e poi chiusi.

È possibile eseguire queste due operazioni, rispettivamente, con le funzioni `fflush` e `fclose`.

`fflush(FILE *stream)`

`fclose(FILE *stream)`



File - Esercizi

- Scrivere un programma in linguaggio C che crea un nuovo file di nome “provaFile.txt” e scrive una stringa data in input dall’utente



File - Esercizi

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

int main()
{
    char str[1000];

    char path[40]="provaFile.txt";

    //Creazione del file
    FILE *file = fopen(path,"w");
    if(file == NULL)
    {
        printf("Errore durante l'apertura del file");
        exit(1);
    }
    printf("Inserire frase da inserire nel file : ");
    fgets(str, sizeof(str), stdin);
    fprintf(file,"%s",str);

    fflush(file);
    fclose(file);
    printf("\n Il file %s è stato creato con successo \n",path);
    return 0;
}
```



File - Esercizi

- Scrivere un programma in linguaggio C che legge il contenuto di un file e conta il numero di righe presenti
 - NB. EOF è un carattere speciale che indica la fine del file



File - Esercizi

```
#include <stdio.h>

int main()
{
    FILE *fptr;
    int numRows = 0;
    char path[100];
    char c;
    printf("Inserire il path del file da aprire : \n");
    scanf("%s", path);

    fptr = fopen(path, "r");
    if (fptr == NULL)
    {
        printf("Impossibile aprire il file %s", path);
        return 0;
    }
    // Lettura del file
    printf("-----\n");
    printf("Contenuto del file: \n");

    while(fscanf(fptr, "%c" , &c) != EOF){
        //Stampo a video il carattere
        printf("%c", c);

        if (c == '\n'){ //Se trovo \n sta iniziando una nuova riga
            numRows = numRows + 1;
        }
    }
    printf("\n-----\n");
    fflush(fptr);
    fclose(fptr);
    printf("Il numero di righe presenti nel file %s è pari a: %d \n", path, numRows+1);
    return 0;
}
```



File - Esercizi

```
#include <stdio.h>

int main()
{
    FILE *fptr;
    int numRows = 0;
    char path[100];
    char c;
    printf("Inserire il path del file da aprire : \n");
    scanf("%s", path);

    fptr = fopen(path, "r");
    if (fptr == NULL)
    {
        printf("Impossibile aprire il file %s", path);
        return 0;
    }
    // Lettura del file
    printf("-----\n");
    printf("Contenuto del file: \n");
    for (c = getc(fptr); c != EOF; c = getc(fptr)){
        //Stampo a video il carattere
        printf("%c", c);

        if (c == '\n'){ //Se trovo \n sta iniziando una nuova riga
            numRows = numRows + 1;
        }
    }
    printf("\n-----\n");
    fflush(fptr);
    fclose(fptr);
    printf("Il numero di righe presenti nel file %s è pari a: %d ", path, numRows+1);
    return 0;
}
```



Domande?

