



# ELEMENTI DI INFORMATICA

DOCENTE: FRANCESCO MARRA

INGEGNERIA CHIMICA

INGEGNERIA ELETTRICA

SCIENZE ED INGEGNERIA DEI MATERIALI

INGEGNERIA GESTIONALE DELLA LOGISTICA E DELLA PRODUZIONE

INGEGNERIA NAVALE


UNIVERSITÀ DEGLI STUDI DI NAPOLI FEDERICO II  
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE

# LIBRERIE E NAMESPACE





# AGENDA

- Programmazione strutturata
    - Librerie utente
  - Namespace
- 



LIBRERIE



# LE LIBRERIE

- Sono raccolte di funzioni riutilizzabili in programmi diversi
- Si ha la sicurezza che le funzioni usate hanno un funzionamento già dimostrato e quindi corretto
  - Il codice può essere organizzato in file differenti
- Programma principale costituito solo dal codice dell'applicazione specifica
  - In testa riferimento a tutte le librerie invocate

# INVOCAZIONE DI UNA LIBRERIA

- Si usa la direttiva *include* per richiamare un file esterno nel file in compilazione
- Due forme sintattiche per specificare al compilatore dove cercare il file
  - **#include <filename>**
    - Il compilatore cerca il file nelle directory scelte dall'ambiente di sviluppo
  - **#include "filename"**
    - Il compilatore cerca il file nella directory corrente o nel percorso specificato

# INTESTAZIONE ED IMPLEMENTAZIONE

- Le librerie sono organizzate in due file distinti
  - File di intestazione con suffisso “.h”
    - Dichiarazione di variabili globali, costanti e tipi
    - Dichiarazione di funzioni
  - File di implementazione o sorgente con suffisso “.cpp” (per C++) o “.c” (per C)
    - Implementazione delle funzioni
- I due file hanno preferibilmente lo stesso nome
- È buona norma inserire anche un file di test con un *main*
  - Verifica la correttezza dell'insieme di funzionalità implementate

# ESEMPIO: LIBRERIA PER LA GESTIONE DI ARRAY MONODIMENSIONALI

- Vector.h

```
/* La libreria definisce struttura ed operazioni su vettori:
   N_MAX      dimensione massima dell'array
   Vector      tipo del vettore
   mean        calcolo media
   max         calcolo massimo
   min         calcolo minimo
*/

// Definisce la dimensione massima dell'array
#define N_MAX 100

// Definisce il tipo Vector come array monodimensionale di reali
typedef double Vector[N_MAX];

//Prototipi delle funzioni della libreria
double mean(Vector, int);
double min(Vector, int);
double max(Vector, int);
```



# ESEMPIO: LIBRERIA PER LA GESTIONE DI ARRAY MONODIMENSIONALI

- Vector.cpp

```
/* Include file di intestazione
della libreria */
#include "Vector.h"

// Implementazione dei prototipi

double mean (Vector v, int n) {
    double mean = 0;
    for (int i = 0; i < n ; i++)
        mean = mean + v[i];
    return mean/n;
}

double max(Vector v, int n) {
    double max = v[0];
    for (int i = 1; i < n ; i++)
        if (max < v[i])
            max = v[i];
    return max;
}
```

```
double min(Vector v, int n) {
    double min = v[0];
    for (int i = 1; i < n ; i++)
        if (min > v[i])
            min = v[i];
    return min;
}
```

# ESEMPIO: LIBRERIA PER LA GESTIONE DI ARRAY MONODIMENSIONALI

- Main.cpp

```
#include <iostream>
#include "Vector.h"

using namespace std;

int main(int argc, char *argv[])
{
    Vector v;
    const int N = 10;

    cout << "Inserisci i " << N << " valori del vettore" << endl << endl;
    for (int i = 0; i < N; i++){
        cout << "Inserisci valore " << i << endl;
        cin >> v[i];
    }

    cout << "\nIl valore medio e': " << mean(v,N) << endl;
    cout << "\nIl valore minimo e': " << min(v,N) << endl;
    cout << "\nIl valore massimo e': " << max(v,N) << endl;

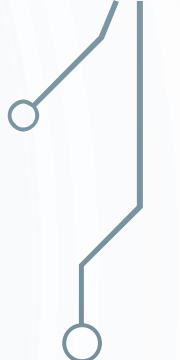
    system("PAUSE");
    return EXIT_SUCCESS;
}
```

# DEV C++ E LE LIBRERIE DEFINITE DALL'UTENTE: UN MODO PRATICO

- Creare un progetto «Console application»
- Dev C++ includerà un file «main.cpp» che sarà il punto di inizio del programma
- Aggiungere le librerie necessarie (file .h e .cpp)
- Compilando il progetto, Dev C++ si occuperà di compilare e «linkare» le librerie e il main automaticamente



# ESERCIZI...

- Provare a risolvere gli esercizi finora svolti con la programmazione strutturata e l'uso di librerie
- 

# NAMESPACE



# NAMESPACE

- Lo spazio globale di definizione di variabili e funzioni può essere suddiviso in diversi contenitori detti *namespace*
- Dichiarazione di *namespace*
  - **namespace <nome> {elenco dichiarazioni/definizioni}**
  - Ad ogni *namespace* è associato un nome ed uno specifico scope

```
namespace Cerchio{  
    // dichiarazione di variabili  
    const double pi = 3.14159;  
  
    //dichiarazione di funzioni  
    double area(double r) {  
        return r * r * pi;  
    }  
    double circonferenza(double r) {  
        return 2 * r * pi;  
    }  
}
```

# ACCESSO A RISORSE INTERNE AD UN NAMESPACE

- Non è richiesta nessuna sintassi particolare per consentire alle funzioni di accedere agli identificatori in esso definiti

```
namespace Cerchio{  
    // dichiarazione di variabili  
    const double pi = 3.14159;  
  
    //dichiarazione di funzioni  
    double area(double r) {  
        return r * r * pi;  
    }  
    double circonferenza(double r) {  
        return 2 * r * pi;  
    }  
}
```

# ACCESSO A RISORSE ESTERNE AD UN NAMESPACE: SOLUZIONE 1

- Qualificazione esplicita mediante l'**operatore di risoluzione di visibilità "::"**

```
int main(int argc, char *argv[])
{
    double raggio;

    cout << "Inserisci raggio " << endl;
    cin >> raggio;

    cout << "\nL'area e': " << Cerchio::area(raggio);

    cout << "\nLa circonferenza e': " << Cerchio::circonferenza(raggio);

    system("PAUSE");
    return EXIT_SUCCESS;
}
```

Il nome del namespace  
ripetuto ogni qualvolta si fa  
riferimento ad un suo  
identificatore



# ACCESSO A RISORSE ESTERNE AD UN NAMESPACE: SOLUZIONE 2

- Qualificazione implicita mediante ***using namespace <nome>***

```
using namespace Cerchio;
```

```
int main(int argc, char *argv[])
{
    double raggio;

    cout << "Inserisci raggio " << endl;
    cin >> raggio;

    cout << "\nL'area e': " << area(raggio);

    cout << "\nLa circonferenza e': " << circonferenza(raggio);

    system("PAUSE");
    return EXIT_SUCCESS;
}
```

I nomi contenuti nel namespace indicato sono utilizzati senza necessità di qualificazione

**DOMANDE, DUBBI, PERPLESSITÀ**

