

**WEB DEVELOPER**  
**Fondamenti di Programmazione**  
**Massimo PAPA**

# **Librerie definite dall'utente**

## **- I MODULI -**

# Esempio

```
#include <iostream>
#include <string>
using namespace std;

double computeTaxes(double income)
{
    if (income<5000) return 0.0;
    return 0.07*(income-5000.0);
}

double getIncome(string prompt)
{
    double reddito;
    cout << prompt;
    cin >> reddito;
    return reddito;
}

void printTaxes(double tasse)
{
    cout << "The taxes is $" << tasse
    << endl;
}
```

```
int main()
{
    // Get the income;
    double reddito;
    double tasse;

    reddito = getIncome("Please enter
        the employee income: ");

    // Compute Taxes
    tasse = computeTaxes(reddito);

    // Print employee taxes
    printTaxes(tasse);

    return 0;
}
```

# Firma della funzione

- La firma della funzione è in realtà simile all'intestazione della funzione tranne per due aspetti:
  - ◆ I nomi dei parametri potrebbero non essere specificati nella firma della funzione
  - ◆ La firma della funzione deve essere terminata con un punto e virgola
- Esempio:

Parametro  
senza nome

Punto e  
virgola

```
double computeTaxes(double) ;
```

# Perché abbiamo bisogno della firma della funzione?

- Per nascondere le informazioni
  - ◆ Se vuoi creare la tua libreria e condividerla con i tuoi clienti senza far conoscere loro i dettagli di implementazione, dovresti dichiarare tutte le firme delle funzioni in un file di intestazione (.h) e distribuire il codice binario del file di implementazione
- Per l'astrazione delle funzioni
  - ◆ Condividendo solo le firme delle funzioni, abbiamo la libertà di modificare i dettagli di implementazione di volta in volta per
    - Migliorare le prestazioni delle funzioni
    - fare in modo che i clienti si concentrino sullo scopo della funzione, non sulla sua implementazione

# Esempio con le firme

```
#include <iostream>
#include <string>
using namespace std;

// Function Signature
double getIncome(string);
double computeTaxes(double);
void printTaxes(double);

void main()
{
    // Get the income;
    double reddito;
    double tasse;

    reddito = getIncome("Please enter
        the employee income: ");

    // Compute Taxes
    tasse = computeTaxes(reddito);

    // Print employee taxes
    printTaxes(tasse);

    return 0;
}
```

```
double computeTaxes(double income)
{
    if (income<5000) return 0.0;
    return 0.07*(income-5000.0);
}

double getIncome(string prompt)
{
    double income;
    cout << prompt;
    cin >> income;
    return income;
}

void printTaxes(double taxes)
{
    cout << "The taxes is $" << taxes <<
        endl;
}
```

# Le librerie utente

- È una buona pratica creare librerie che possano essere utilizzate da te e dai tuoi clienti
- Per creare librerie C++, devi
  - ◆ Creare il file di intestazione per memorizzare le firme delle funzioni
  - ◆ Creare il file di implementazione per memorizzare il codice delle funzioni
  - ◆ Includere il file di intestazione nel programma per utilizzare le funzioni definite dall'utente

# File di intestazione C++

- I file di intestazione C++ devono avere l'estensione .h e dovrebbero avere la seguente struttura
  - ◆ #ifndef direttiva del compilatore
  - ◆ #define direttiva del compilatore
    - ◆ Può includere altri file di intestazione
    - ◆ Tutte le funzioni sono firmate con alcuni commenti su cosa fanno, i parametri formali e il valore di ritorno
  - ◆ #endif direttiva del compilatore

# File di intestazione TaxesRules

```
#ifndef _TAXES_RULES_
#define _TAXES_RULES_

#include <iostream>
#include <string>
using namespace std;

/* scopo -- ottenere il
   reddito dei dipendenti
   input -- un prompt di
   stringa da mostrare
   all'utente
   output -- un valore double
   che rappresenta il
   reddito
*/
double getIncome(string);
```

```
/* scopo -- calcolare le
   tasse per un dato reddito
   input -- un valore double
   che rappresenta il
   reddito
   output -- un valore double
   che rappresenta le tasse
*/
double computeTaxes(double);

/* scopo -- mostrare le
   tasse all'utente
   input -- un valore double
   che rappresenta le tasse
   output -- Nessuno
*/
void printTaxes (double);

#endif
```



# File di implementazione di TaxesRules

```
#include "TaxesRules.h"
```

```
double computeTaxes (double  
    reddito)  
{  
    if (reddito<5000) return  
        0.0;  
    return  
        0.07*(reddito-5000.0);  
}
```

```
double getIncome (string  
    prompt)  
{  
    double reddito;  
    cout<< prompt;  
    cin >> reddito;  
    return reddito;
```

```
void printTaxes (double  
    tasse)  
{  
    cout << "Le tasse sono $"  
        << tasse << endl;  
}
```

# Programma principale

```
#include "TaxesRules.h"
int main(){
    // Ottieni il reddito;
    double reddito;
    double tasse;

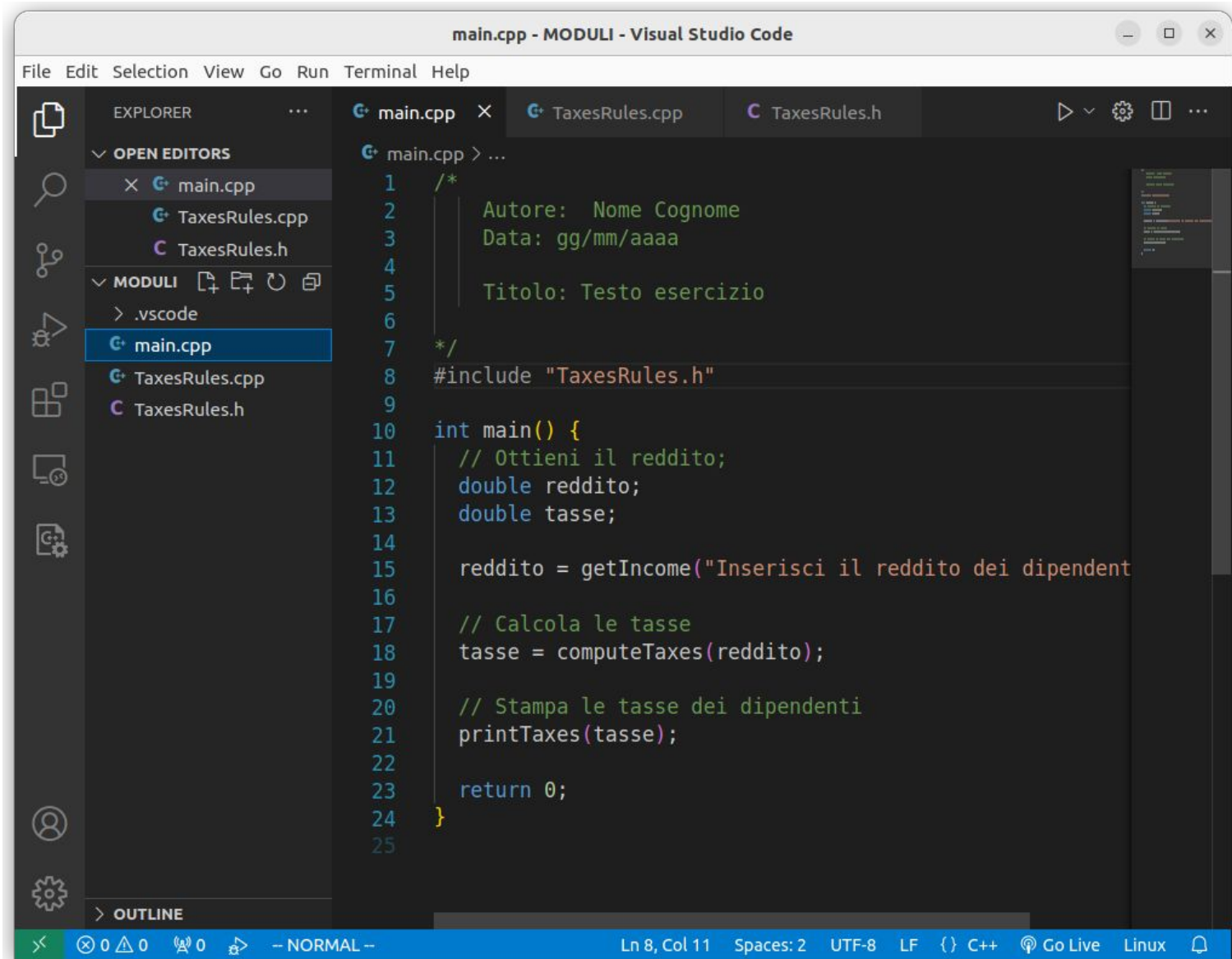
    Reddito = getIncome("Inserisci il reddito dei
        dipendenti: ");

    // Calcola le tasse
    tasse = computeTaxes(reddito);

    // Stampa le tasse dei dipendenti
    printTaxes(tasse);

    return 0;
```

# VSCoCode – main.cpp



main.cpp - MODULI - Visual Studio Code

File Edit Selection View Go Run Terminal Help

EXPLORER

OPEN EDITORS

- main.cpp
- TaxesRules.cpp
- TaxesRules.h

MODULI

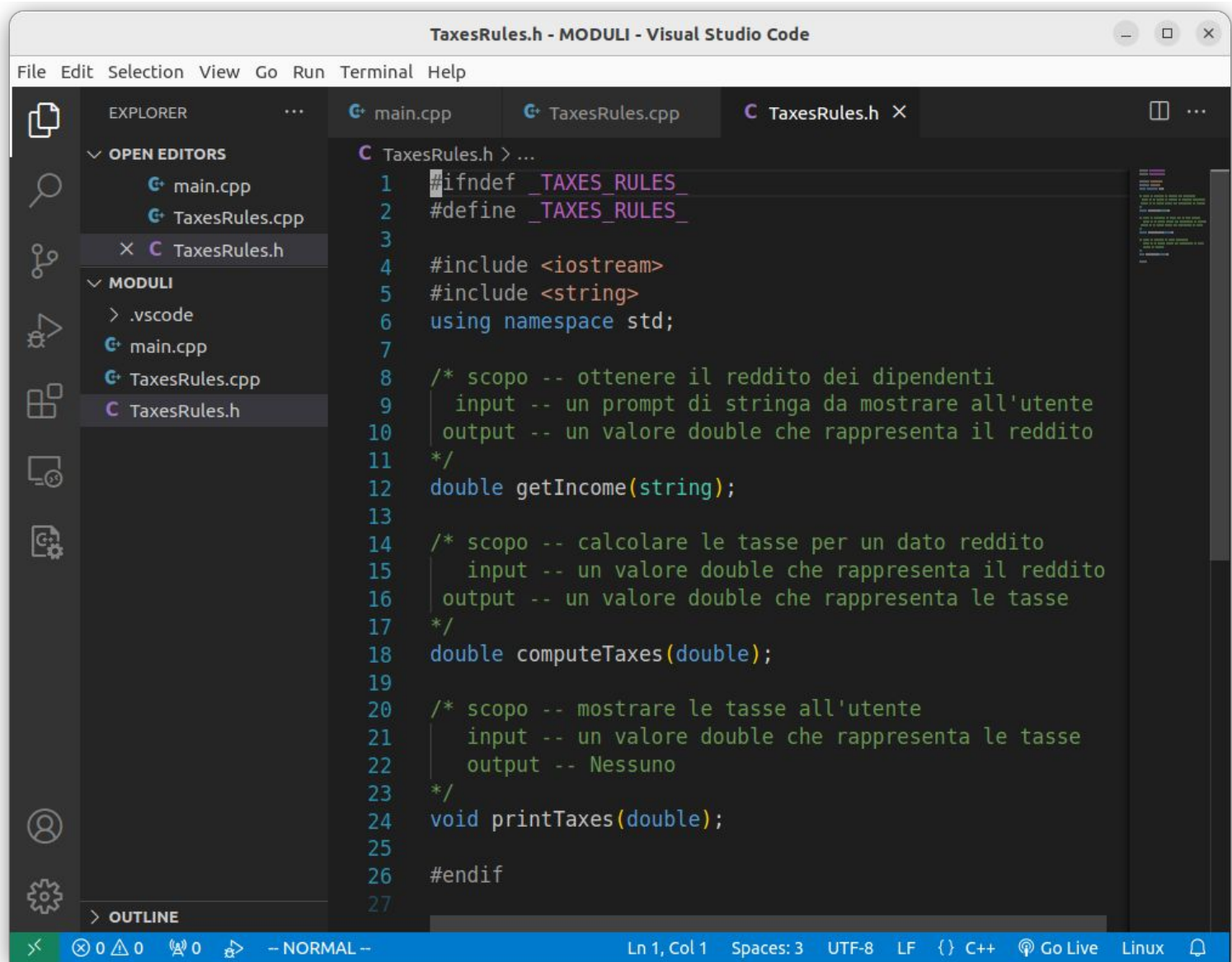
- .vscode
- main.cpp
- TaxesRules.cpp
- TaxesRules.h

main.cpp > ...

```
1  /*
2      Autore:  Nome Cognome
3      Data:   gg/mm/aaaa
4
5      Titolo: Testo esercizio
6
7  */
8  #include "TaxesRules.h"
9
10 int main() {
11     // Ottieni il reddito;
12     double reddito;
13     double tasse;
14
15     reddito = getIncome("Inserisci il reddito dei dipendenti");
16
17     // Calcola le tasse
18     tasse = computeTaxes(reddito);
19
20     // Stampa le tasse dei dipendenti
21     printTaxes(tasse);
22
23     return 0;
24 }
25
```

Ln 8, Col 11 Spaces: 2 UTF-8 LF {} C++ Go Live Linux

# VSCoDe – TaxesRules.h



TaxesRules.h - MODULI - Visual Studio Code

File Edit Selection View Go Run Terminal Help

EXPLORER

OPEN EDITORS

- main.cpp
- TaxesRules.cpp
- TaxesRules.h

MODULI

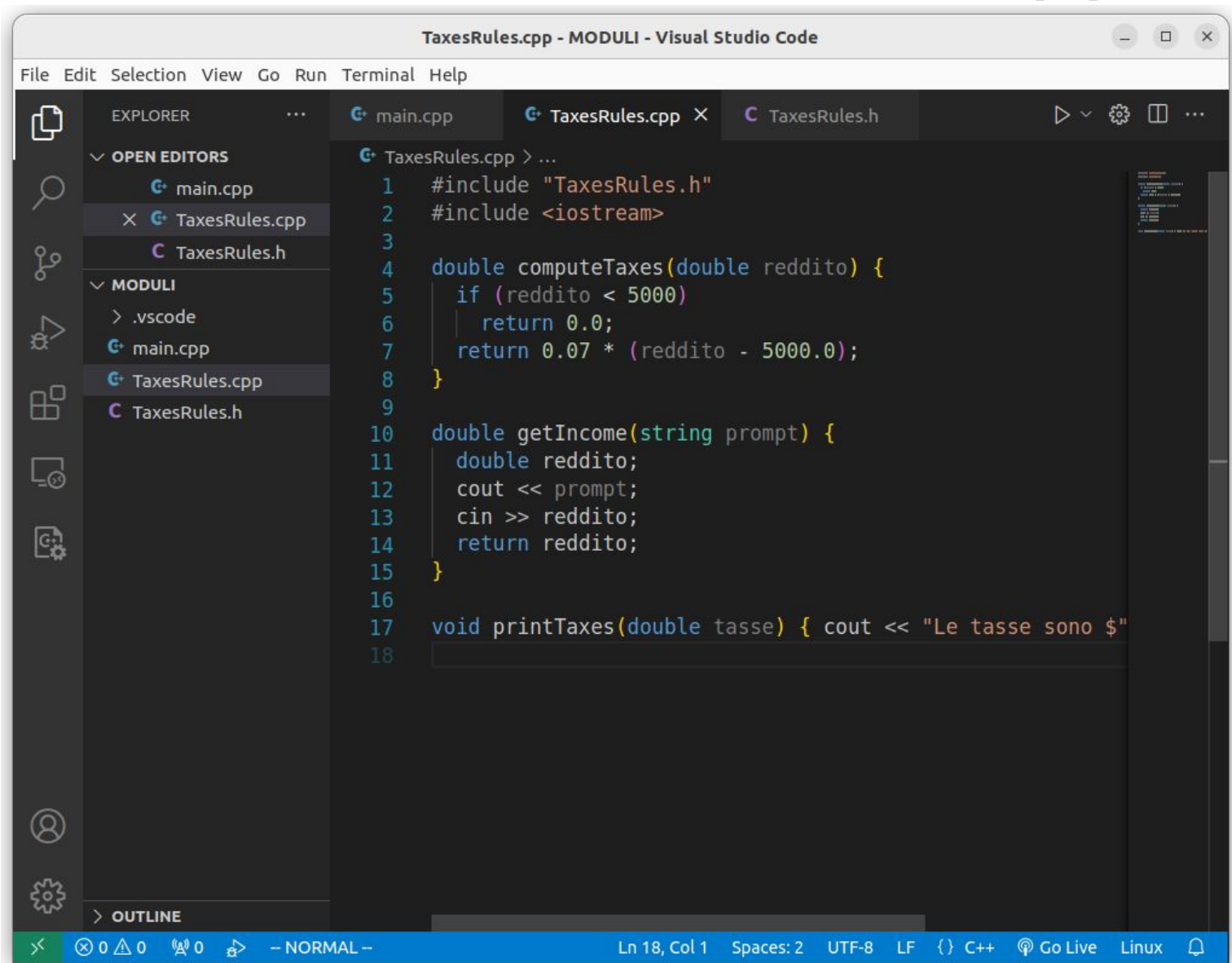
- .vscode
- main.cpp
- TaxesRules.cpp
- TaxesRules.h

OUTLINE

```
1 #ifndef _TAXES_RULES_
2 #define _TAXES_RULES_
3
4 #include <iostream>
5 #include <string>
6 using namespace std;
7
8 /* scopo -- ottenere il reddito dei dipendenti
9  input -- un prompt di stringa da mostrare all'utente
10 output -- un valore double che rappresenta il reddito
11 */
12 double getIncome(string);
13
14 /* scopo -- calcolare le tasse per un dato reddito
15  input -- un valore double che rappresenta il reddito
16  output -- un valore double che rappresenta le tasse
17 */
18 double computeTaxes(double);
19
20 /* scopo -- mostrare le tasse all'utente
21  input -- un valore double che rappresenta le tasse
22  output -- Nessuno
23 */
24 void printTaxes(double);
25
26 #endif
27
```

Ln 1, Col 1 Spaces: 3 UTF-8 LF {} C++ Go Live Linux

# VSCode – TaxesRules.cpp



The screenshot shows the Visual Studio Code interface with the file TaxesRules.cpp open. The Explorer sidebar on the left shows the project structure with files main.cpp, TaxesRules.cpp, and TaxesRules.h. The main editor area displays the code for TaxesRules.cpp, which includes headers, a computeTaxes function, a getIncome function, and a printTaxes function. The status bar at the bottom indicates the current position is Line 18, Column 1, with 2 spaces, UTF-8 encoding, and LF line endings.

```
TaxesRules.cpp - MODULI - Visual Studio Code
File Edit Selection View Go Run Terminal Help

EXPLORER
  OPEN EDITORS
    main.cpp
    TaxesRules.cpp
    TaxesRules.h
  MODULI
    .vscode
    main.cpp
    TaxesRules.cpp
    TaxesRules.h

TaxesRules.cpp > ...
1  #include "TaxesRules.h"
2  #include <iostream>
3
4  double computeTaxes(double reddito) {
5      if (reddito < 5000)
6          return 0.0;
7      return 0.07 * (reddito - 5000.0);
8  }
9
10 double getIncome(string prompt) {
11     double reddito;
12     cout << prompt;
13     cin >> reddito;
14     return reddito;
15 }
16
17 void printTaxes(double tasse) { cout << "Le tasse sono $"
18
```

Ln 18, Col 1 Spaces: 2 UTF-8 LF {} C++ Go Live Linux

# VSCode – tasks.json

originale :

```
{
  "tasks": [
    {
      "type": "cppbuild",
      "label": "C/C++: g++ build active file",
      "command": "/usr/bin/g++",
      "args": [
        "-fdiagnostics-color=always",
        "-g",
        "${file}",
        "-o",
        "${fileDirname}/${fileBasenameNoExtension}"
      ],
      "options": {
        "cwd": "${fileDirname}"
      },
      "problemMatcher": [
        "$gcc"
      ],
      "group": {
        "kind": "build",
        "isDefault": true
      },
      "detail": "Task generated by Debugger."
    }
  ],
  "version": "2.0.0"
}
```

modificato :

```
{
  "tasks": [
    {
      "type": "shell",
      "label": "C/C++: g++ build active file",
      "command": "/usr/bin/g++",
      "args": [
        "-fdiagnostics-color=always",
        "-g",
        "${fileDirname}/*.cpp",
        "-o",
        "${fileDirname}/${fileBasenameNoExtension}"
      ],
      "options": {
        "cwd": "${fileDirname}"
      },
      "problemMatcher": [
        "$gcc"
      ],
      "group": {
        "kind": "build",
        "isDefault": true
      },
      "detail": "Task generated by Debugger."
    }
  ],
  "version": "2.0.0"
}
```

Intervento realizzato da