



UNIVERSITÀ  
DEGLI STUDI  
FIRENZE



# Esercitazione

Exception + STL exercise

# Obiettivo

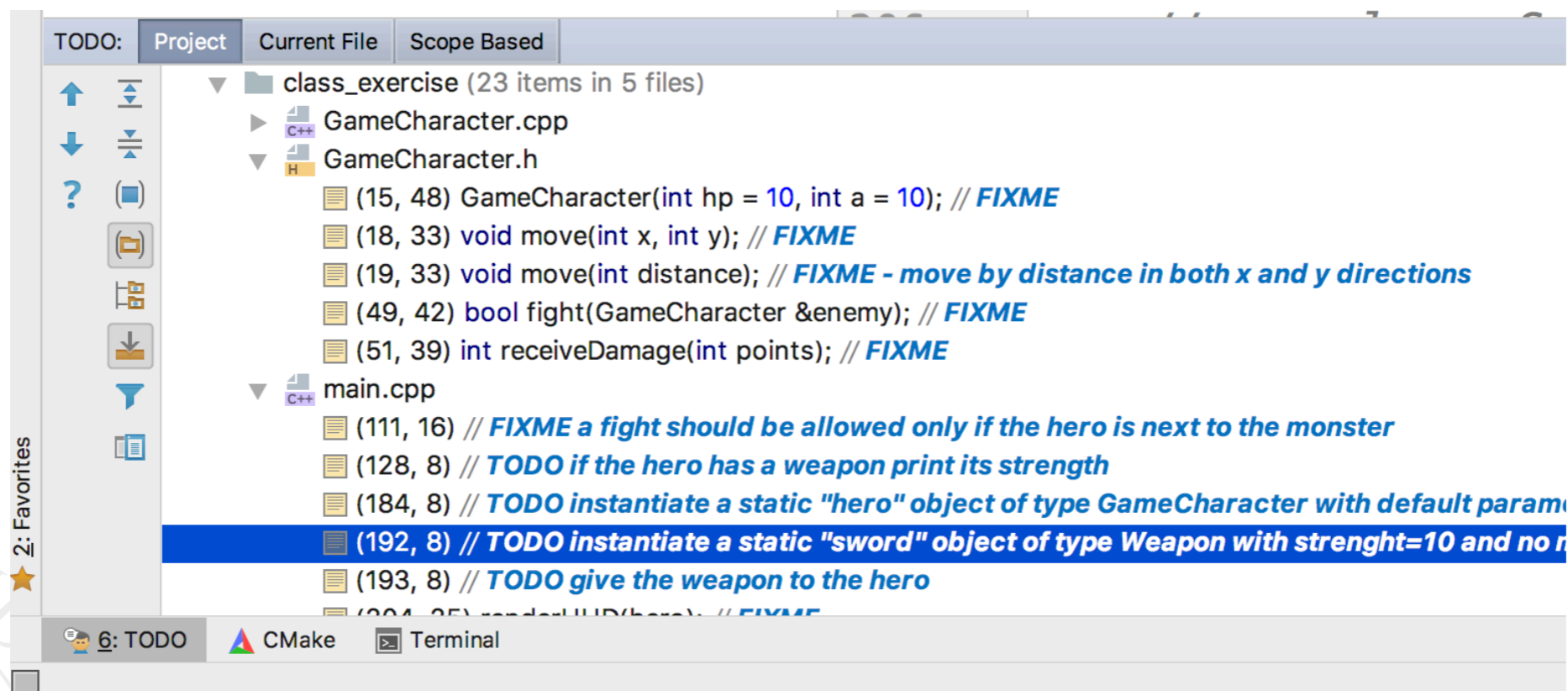
- Il progetto CLion fornito contiene classi e scheletri di classi relative al gioco in stile Rogue (<https://it.wikipedia.org/wiki/Roguelike>) della scorsa esercitazione.
- Scopo della presente esercitazione è:
  - Implementare una classe `GameFileException` per rappresentare eccezioni relative a problemi con file necessari al funzionamento del gioco
    - Es. file mancanti, non leggibili o non decomprimibili
  - Lanciare e prendere eccezioni, decidendo a quale livello di stack delle chiamate intervenire. Gestire gerarchia di ereditarietà di eccezioni.
  - Dichiarare un vector STL capace di contenere diversi tipi di mostri (ovvero gestire oggetti polimorfici), fare semplici operazioni (es. passaggio come argomento, iterazione di elementi, inserimento di elementi)

# Schema del codice

- Lo schema del codice delle classi di base è lo stesso di quello della volta scorsa. In particolare opereremo su:
- `Dungeon` crea mappe casuali con stanze, corridoi, scale, porte, etc. Dovrà lanciare eccezioni nelle funzioni di salvataggio/caricamento mappa
- `GameFileException` rappresenta un'eccezione relativa a problemi con file. Deve essere completata e deve estendere una classe base di eccezione
- `FakeJPEG` simula la lettura di immagini JPEG. Deve essere implementata per gestire possibili problemi di lettura di file.
- `Main` nel file andremo a modificare una funzione e parti del `main()` per usare vettori STL e gestire eccezioni su file
- `splash_screen` funzioni per mostrare uno splash screen da immagini JPEG (simulate). Dovremo gestire possibili problemi di caricamento delle immagini.

# Dove modificare il codice

- Le indicazioni precise sul codice da modificare sono fornite come commenti indicati con TODO e FIXME
- Per vedere tutti questi commenti selezionare la finestra TODO di CLion



# Dove modificare il codice

The screenshot shows the CLion IDE interface. The main editor displays the code in `main.cpp` with the following content:

```
185 // find a legal start position
186 int startX = 0;
187 int startY = 0;
188 setupCharacterCell(startX, startY, map);
189 hero.setPosX(startX);
190 hero.setPosY(startY);
191 // create a weapon and give it to hero
192 // TODO instantiate a static "sword" object of type Weapon with strenght=10 and
193 // TODO give the weapon to the hero
194 // create an enemy with a low grade armor
195 GameCharacter enemy(20, 2);
196 // find monster position not too far from hero position
197 startX += 5;
198 startY += 3;
199 setupCharacterCell(startX, startY, map);
200 enemy.setPosX(startX);
201 enemy.setPosY(startY);
202
203 // render
204 renderHUD(hero); // FIXME
205 renderGame(map, hero, enemy);
```

The TODO list at the bottom of the IDE shows the following items:

- (15, 48) `GameCharacter(int hp = 10, int a = 10);` // FIXME
- (18, 33) `void move(int x, int y);` // FIXME
- (19, 33) `void move(int distance);` // FIXME - move by distance in both x and y directions
- (49, 42) `bool fight(GameCharacter &enemy);` // FIXME
- (51, 39) `int receiveDamage(int points);` // FIXME
- (111, 16) // FIXME a fight should be allowed only if the hero is next to the monster
- (128, 8) // TODO if the hero has a weapon print its strength
- (184, 8) // TODO instantiate a static "hero" object of type GameCharacter with default parameters
- (192, 8) // TODO instantiate a static "sword" object of type Weapon with strenght=10 and no magic
- (193, 8) // TODO give the weapon to the hero
- (204, 25) `renderHUD(hero);` // FIXME

The status bar at the bottom indicates the current context is `class_exercise [D]`.

# Classe GameFileException

- Estendere la classe base `std::runtime_error`.
- Aggiungere attributi utili per portare informazioni sui file con problemi, come il nome del file ed il fatto se l'errore non è recuperabile e debba portare alla terminazione del programma.
- Implementare il costruttore; invocare il costruttore di classe base.





# Classe Dungeon/FakeJPEG

- Lanciare diverse tipologie di eccezioni come indicato.





# main/splash\_screen

- Decidere come e dove gestire le eccezioni lanciate da FakeJPEG e Dungeon quando si leggono/scrivono i file





# main

- Istanziare un vector STL in grado di contenere diversi tipi di mostri.
- Implementare la funzione createMonsters per riempire il vettore con mostri in posizioni casuali della mappa
- Implementare la funzione checkMonsterPosition per iterare su tutti i mostri del vettore