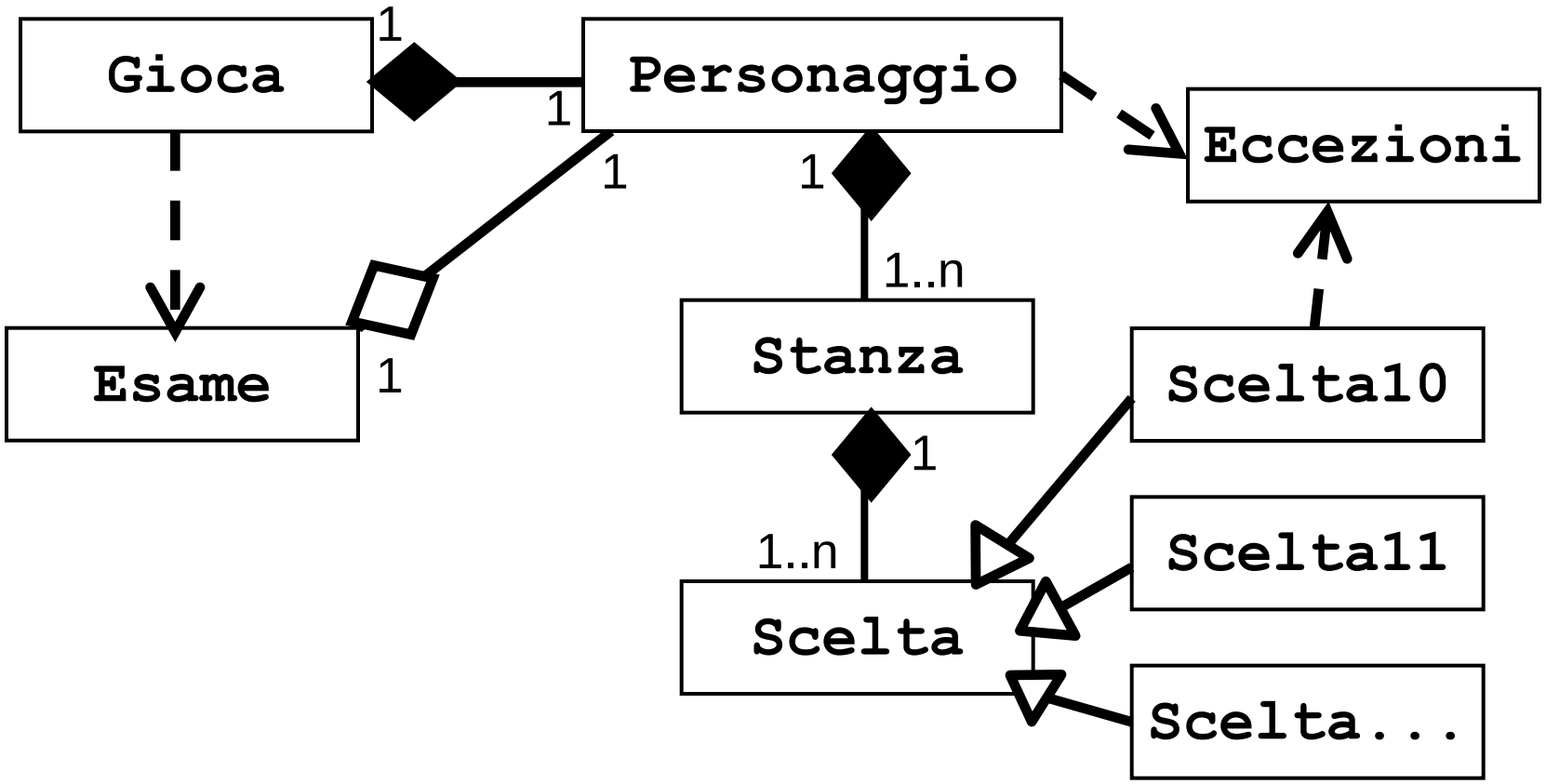
Progetto per il corso

- Progettare un simulatore di attività.
  - Schema della trama: realizzare un obiettivo entro un certo “tempo” (contato in giorni/turni/mesi...)
- Il “tempo” può essere impiegato per:
  - potenziare il personaggio
  - inseguire l’obiettivo
  - comprare oggetti

Le classi del modulo

- La struttura relazionale delle classi nell'esempio.



La classe `Personaggio`

- Contiene tutti i dati rilevanti del giocatore: ad esempio, attributi ed energie di tipo intero.
- Inoltre ha un riferimento alla stanza corrente.
  - e anche un oggetto **`Random`** per “tirare dadi”
- Può essere ampliata con attributi particolari di tipo diverso da “numerico intero”
  - attributi frazionari (double)
  - inventario degli oggetti trasportati (non per forza una collezione, anche se è ragionevole)
  - ...

La classe **`Stanza`**

- Contiene un **`nome`** e un array di **`Scelta`**.
  - La dimensione è variabile.
  - Ogni **`Scelta`** rappresenta un'azione possibile: può avere un tipo dinamico sottoclasse (da vedere dopo).
  - La classe base ha un unico metodo **`azione`**, che comanda di spostare il personaggio ad un'altra stanza
- La creazione delle stanze:
  - viene da una mappa complicata
  - viene creato un array che poi si perde, ma la prima stanza → **`dove`** del personaggio

Polimorfismo delle scelte

- La classe **`Scelta`** serve come classe base, per la navigazione nei menù, ma ci saranno anche classi estese **`Scelta10`**, **`Scelta11`**, ...
  - basta sovrascrivere l'unico metodo **`azione`**
  - grazie all'early binding, ogni stanza possiede sempre un array di oggetti di tipo **`Scelta`**
  - ma con il late binding l'effetto di ogni scelta è diverso (esempi tipici: azioni che spostano i punteggi, azioni condizionali...)

Progetto dell'avventura

- Per progettare il gioco occorre:
  - mappa delle stanze/menù
  - azioni per ogni stanza (possibilmente cercando di bilanciare gli effetti)
  - eventi speciali del gioco (esempio: l'esame finale)
- Poi, implementare tutti i metodi.
- Debug a due livelli:
  - correttezza semantica del programma
  - bilanciamento del gioco (alpha-testing)

Estensioni

È senz'altro necessario fare espansioni. Esempi:

- Gestione delle eccezioni
  - eventi indesiderati possono essere controllati
- Salvataggio e caricamento da disco
- Lettura più efficiente dei messaggi e dei menù utente (liste, lettura dei dati da file)
- Azioni in tempo reale o quiz:
  - creare un "gioco nel gioco"
  - temporizzare azioni

Estensioni

- Item (oggetti): dotati di nome, caratteristiche
  - implementare un inventario
  - prevedere effetti diversi dati dagli oggetti
- Persone: estensione di Item
  - inserire opzioni di dialogo più o meno complesse
  - inserire rapporti con personaggi non giocanti
- Evoluzione del personaggio
  - classe estesa del personaggio precedente

Estensioni

- Inserire thread multipli:
  - eventi che capitano dopo un certo ammontare di tempo reale
  - personaggi mobili (o in tempo reale, o in un certo numero di mosse)
- Inserire grafica
  - basta in realtà integrare le stanze con un’immagine e rimpiazzare la stampa testuale del menù (array di scelte) con uno grafico (a bottoni)