linea orizzontale

**Bisignano Simone Matricola: 20031305**

**Pezzano Paolo Raffaele Matricola: 20024419**

Uniupo

Disit Alessandria

Relazione Agenda Java

# INTRODUZIONE

Per questo progetto,l’obiettivo principale è stato quello di sviluppare un’agenda per gestire gli appuntamenti. In particolare, essa dovrà offrire determinate opzioni come ad esempio l’inserimento,la rimozione e/o cancellazione di un particolare appuntamento, memorizzando i dati all’interno di un file di testo accessibile e modificabile dall’utente. Tramite un semplice menù, composto da una serie di operazioni ammesse all’interno dell’agenda, l’utente potrà gestire la propria lista di appuntamenti.

# OBIETTIVI

1. Inserire un appuntamento.
2. Eliminare un appuntamento.
3. Modificare un appuntamento.
4. Effettuare ricerca di appuntamenti per data singola
5. Effettuare ricerca di appuntamenti indicando un range di fate
6. Effettuare ricerca di appuntamenti indicando il nome
7. Stampare tutta l’agenda in ordine cronologico
8. Creare un’interfaccia per l’agenda
9. Effettuare i test per l’agenda.

# SVOLGIMENTO

Per prima cosa, ci siamo occupati di gestire gli orari dei vari appuntamenti. Abbiamo trattato principalmente il formato dell’orario, poichè certi valori non sono validi, quindi abbiamo controllato che l’utente abbia inserito un’ora valida per l’appuntamento. Successivamente, dopo aver creato l’oggetto di tipo Orario, abbiamo calcolato anche la durata di fine dell'appuntamento, mediante il metodo “fineOrario” che prende in input una durata inserita dall’utente.

E’ presente inoltre, un metodo particolare che controlla che la prima data inserita sia antecedente alla seconda, in modo da evitare errori di forma nell’agenda.

Dopo aver completato la classe Orario, abbiamo svolto la nuova classe Data. Anch’essa, deve controllare come prima cosa l’input dell’utente, ossia controllare che il formato di essa (gg-mm-aaaa) sia correttamente inserito. Tramite il costruttore, abbiamo separato i cambi sfruttando il metodo “split”(dato che l’utente inserisce una stringa), grazie al quale utilizzando un particolare carattere, siamo riusciti a suddividere il giorno, la data e l’anno. Dopo aver effettuato il controllo, abbiamo effettivamente creato l’oggetto data. Successivamente, ci siamo occupati della classe Appuntamento, la quale contiene più costruttori in base all’input inserito dall’utente ed altri molteplici metodi che ci hanno permesso di completare gli obiettivi elencati precedentemente. Innanzitutto, tramite il metodo “checkInputAppuntamento” abbiamo controllato i valori inseriti dall’utente,in modo tale che venga rispettato ogni formato predefinito nelle altre classi, controllando quindi : data, orario, nome, luogo ed anche la durata, la quale non può sforare le 24 ore (1440=60 minuti\*24). Dopo aver completato questa piccola parte, ci siamo occupati di creare il metodo “checkAppuntamento”,il quale stabilisce quale dei due appuntamenti viene cronologicamente prima rispetto all’altro. Abbiamo creato inoltre, il metodo che permette all’utente di modificare la durata dell’appuntamento, quest’ultimo non dovrà superare le 24 ore giornaliere, altrimenti verrà arrotondato alle 23:59. A catena, abbiamo creato anche i getters e setters di data, tramite il metodo “setData” l’utente è in grado di modificare la data dell’appuntamento in base ai propri impegni.

Come penultimo obiettivo, abbiamo realizzato la classe Agenda che ovviamente si occupa di gestire tutti gli appuntamenti inseriti dall’utente in modo totalmente trasparente ad esso. L’utente può gestire la propria agenda tramite file di testo su cui verranno memorizzato gli appuntamenti da lui inseriti, così da poterli visualizzare una volta completato l’inserimento e chiusa l’agenda.

Infine,abbiamo creato la classe Gestione Agenda che rappresenta tramite un menù a scelta, le varie operazioni possibili all’interno di essa. Tramite uno switch (costrutto), abbiamo gestito ogni operazione separatamente, richiamando il metodo opportuno per le varie scelte. Inoltre abbiamo gestito le varie eccezioni con try e catch per eventuali errori che potevano presentarsi all’inserimento di un input, utilizzando quindi OraException e DataException i quali vengono chiamati per sollevare le eccezioni.