FACETS

Realizzare un sistema che simuli il funzionamento della web app FACETS, realizzata nell'ambito del progetto di ricerca europeo omonimo. L'obiettivo del progetto è quello di analizzare, attraverso l'uso dell'intelligenza artificiale, il modo in cui gli iscritti utilizzano le foto di profilo sui propri account social. Tutte le classi, ad eccezione della classe `Esempio`, si devono trovare nel package `facets`.

R1. Iscritti

Il sistema, rappresentato da un oggetto di classe `Facets`, permette di collezionare ed elaborare le immagini di profilo dei donatori che partecipano alla ricerca. La registrazione di un nuovo iscritto, rappresentato dalla classe `Iscritto`, nel sistema avviene attraverso il metodo **nuovaIscrizione()** della classe `Facets`, che riceve come parametri l'email (di cui va garantita l'univocità), il nome, il cognome, il genere e l'età della persona che intende iscriversi. Il metodo si occupa di registrare il nuovo iscritto nel sistema restituendone il riferimento, oppure di restituire il riferimento all'oggetto precedentemente creato in caso di email già presente nel sistema. La classe `Iscritto` offre opportuni metodi per accedere alle suddette informazioni.

Il metodo **descriviIscritto()** della classe `Facets` riceve come parametro l'email di un iscritto e, in caso di iscritto già esistente, restituisce una stringa riportante i campi email, nome, cognome, genere ed età separati da uno spazio.

Il metodo **cercaIscrittoPerEmail()** riceve come parametro l'email di un iscritto e restituisce l'oggetto `Iscritto` corrispondente (`null` se inesistente). Il metodo **cercaIscrittoPerNomeCognome()** restituisce invece un elenco di iscritti per i quali nome e cognome corrispondono a quelli passati come parametri (in ordine alfabetico, con eventuali omonimi in ordine di iscrizione). Quest'ultimo metodo ammette ricerche parziali, in cui possono venir passati come parametri anche solo parte del nome e del cognome. Il metodo **elencoIscritti()** restituisce un elenco di tutti gli iscritti (con il medesimo ordinamento).

R2. Immagini di Profilo

Le immagini sono rappresentate da oggetti di classe `Immagine` e si dividono in diverse categorie sulla base del social network in cui sono utilizzate dalla persona come immagine di profilo; in particolare, il sistema permette di donare immagini utilizzate su Facebook, Instagram o su un altro generico social network, rappresentate rispettivamente dalle classi `ImmagineFacebook`, `ImmagineInstagram` e `ImmagineAltroSocial`. Per donare una nuova immagine si utilizza il metodo **nuovaImmagine()** della classe `Facets`, che riceve come parametri l'email dell'iscritto che dona l'immagine ed una stringa che identifica il nome dato all'immagine dall'iscritto stesso. Se tale iscritto esiste, il metodo crea una nuova immagine e le assegna un codice numerico incrementale a partire da `1` (univoco a prescindere dalla categoria di immagine). Esistono tre versioni del metodo, ognuna delle quali crea un oggetto appartenente a una delle categorie precedentemente menzionate. In caso di immagini di categoria `ImmagineFacebook`, il metodo riceve una stringa identificante il nome dell'album in

cui l'immagine è salvata su Facebook. In caso di immagini di categoria `ImmagineInstagram`, il metodo riceve un valore intero indicante il numero di like ricevuti dall'immagine su Instagram. Infine, in caso di immagini di categoria `ImmagineAltroSocial`, il metodo riceve una stringa identificante il nome del social network in cui tale immagine è utilizzata e un valore `double` rappresentante la dimensione in MB dell'immagine stessa. In ciascun caso, il metodo restituisce un riferimento all'oggetto creato. La classe `Immagine`, così come le classi `ImmagineFacebook`, `ImmagineInstagram` e `ImmagineAltroSocial`, mettono a disposizione opportuni metodi per accedere alle suddette informazioni.

La classe `Facets` mette a disposizione un metodo **descriviImmagine()** che riceve il codice di un'immagine e restituisce una stringa riportante le informazioni relative a tale immagine (se esistente) separate da spazio nell'ordine codice immagine, email dell'iscritto, nome dell'immagine, tipo di immagine (nelle forme "FACEB", "INSTA", "ALTRO") e l'attributo o gli attributi specifici del particolare tipo di immagine (ad esempio, nel caso di immagini di categoria `ImmagineAltroSocial` vengono riportati il nome del social network e la dimensione dell'immagine in quest'ordine).

Attraverso il metodo **cercaImmagine()** della classe `Facets` è possibile cercare un'immagine precedentemente definita. Il metodo riceve come parametro il codice di un'immagine e, in caso di immagine trovata, restituisce il riferimento all'oggetto corrispondente (`null` se non trovata).

È possibile, inoltre, ottenere collezioni di immagini ordinate secondo criteri diversi. In particolare, il metodo **elencoImmaginiPerCodice()** restituisce un elenco di immagini ordinate in modo crescente rispetto al codice identificativo dell'immagine. Il metodo **elencoImmaginiPerTipologia()** restituisce invece una collezione di immagini che riporta prima le immagini di categoria `ImmagineFacebook`, poi quelle di categoria `ImmagineInstagram`, ed infine quelle di categoria `ImmagineAltroSocial`; le immagini di categoria `ImmagineInstagram` vengono riportate in ordine decrescente rispetto al numero di like. Infine, il metodo **elencoImmaginiAltroSocialPerDimensione()** restituisce un elenco delle immagini che vengono da un altro social network ordinate per dimensione crescente.

R3. Analisi delle immagini con l'IA

Il sistema si occupa di creare per gli iscritti dei report personalizzati riportanti le informazioni individuate dall'IA nelle immagini. A tal fine, si utilizza il metodo **nuoviReport()**, che riceve come parametri l'email di un iscritto e una collezione di stringhe che rappresentano le descrizioni di alcune delle immagini di quell'iscritto. Tali stringhe sono formate da tre campi separati dal carattere `";"`. Il primo campo riporta il codice di un'immagine, il secondo una descrizione testuale dell'immagine come elaborata dall'IA e il terzo un timestamp nel formato `AAAAMMGG HH:MM:SS` che rappresenta il momento in cui il report per tale immagine viene generato. Nel caso in cui l'iscritto esista, il metodo verifica l'esistenza delle immagini il cui codice è riportato nelle descrizioni passate come parametro. Per ciascuna delle immagini esistenti e appartenenti a tale iscritto, nel caso in cui il nuovo timestamp sia successivo a quello registrato per l'ultimo report su tale immagine (o in caso di primo report per quell'immagine) il metodo crea un nuovo report su quell'immagine, rappresentato da un oggetto di classe `Report`. Il metodo assegna al report un codice numerico incrementale a

partire da `1`. Si consideri l'eventualità che per ogni immagine possa esistere un numero non definito a priori di report con timestamp diversi tra loro. Il metodo restituisce una collezione di report contenente i report creati sulla base dei criteri appena descritti, in ordine di creazione. In caso di un iscritto inesistente il metodo non sortisce alcun effetto (ed il valore di ritorno è `null`).

Il metodo **reportImmagine()** della classe `Facets` riceve come parametro il codice di un'immagine e restituisce una collezione contenente i report associati a quella particolare immagine per timestamp crescenti (`null` in caso di immagine inesistente).

Il metodo **cercaReportPerId()** riceve come parametro il codice di un report e, in caso di report esistente, restituisce il riferimento all'oggetto corrispondente. Nel caso in cui il codice passato come parametro non corrisponda ad alcun report, il metodo scatena un'eccezione di tipo `ReportNonEsistenteException.`

È possibile infine elencare i report associati a un determinato iscritto. A tal fine, il metodo **elencoReportPerEmailIscritto()** riceve come parametro l'email di un iscritto e restituisce una collezione di report ad esso associati ordinata per timestamp consecutivi. Nel caso in cui l'email passata come parametro non corrisponda ad alcun iscritto, verrà scatenata un'eccezione di tipo `IscrittoNonEsistenteException.`

R4. Lettura dati da file

Il metodo **leggiDatiFacets()** della classe `Facets` permette di leggere da file le informazioni relative agli iscritti ed alle immagini gestite dal sistema. Il metodo riceve come parametro il percorso di un file di testo organizzato per righe, nel quale ogni riga può riportare le informazioni relative ad un iscritto o ad un'immagine.

Le righe relative ad un iscritto iniziano con la stringa "ISC", seguita da email, nome, cognome, genere ed età. Le righe relative alle immagini iniziano invece con la stringa "IMM"; la stringa iniziale sarà seguita da email dell'iscritto, nome dell'immagine, tipologia dell'immagine ("FACEB", "INSTA" oppure "ALTRO") e dagli attributi specifici del tipo di immagine.

Gli elementi di ciascuna riga sono separati gli uni dagli altri tramite il carattere "`;`".

Si assuma che il programma debba gestire eventuali errori di formato del file.