Siti web

Scritto dell'Appello di Programmazione Febbraio 2019 (Java)

Introduzione

In questo esame vi chiederemo di scrivere un insieme di classi per che riguardano siti web, pagine web ecc.

Classi

Descriviamo ora nei dettagli come realizzare le classi necessarie al progetto. Se lo ritenete, potete apportare cambiamenti alle segnature dei metodi proposti e anche – se lo ritenete – alla struttura delle classi, restando ovviamente all'interno dei requisiti esposti nel paragrafo introduttivo.

N.B. (1): Oltre ai metodi qui indicati, potete aggiungere ad ogni classe un valido metodo `toString()` e, dove lo ritenete, anche un metodo `equals` e un metodo `hashCode` (con le segnature appropriate e che soddisfino i contratti previsti).

N.B. (2): Nel seguito, useremo il termine *lista di X* per indicare un qualunque tipo che consenta di conservare delle sequenze di oggetti di tipo `X`: potete implementare una lista con un array (`X[]`) oppure usando ad esempio delle `ArrayList<X>` (pacchetto `java.util`).

Classe URL

Un'URL è una stringa con un formato fisso che, per le esigenze di questo esame, possiamo supporre essere il seguente:

```
PROT://HOST/PATH
```

seguita opzionalmente da:

- una parte chiamata query della forma

  ?QUERY

- una parte chiamata frammento della forma

#FRAGM

dove `PROT` (il protocollo) è una stringa costituita da caratteri alfabetici, `HOST` (lo host) è una stringa costituita da caratteri alfanumerici e da punti, `PATH` è una stringa contenente caratteri alfanumerici, punti e /, `QUERY` è costituita da qualunque carattere che non sia un punto interrogativo o un #, e infine `FRAMM` è costituito da caratteri alfanumerici.

Ad esempio

```
http://pippo.xxx.it/ciao/t3/viao/tron?c=7;d=ciao
```

ha

- il protocollo è `http`
- lo host è `pippo.xxx.it`
- il path è `ciao/t3/viao/tron`
- la query è presente ed è `c=7;d=ciao`
- il frammento non è presente.

Scrivete una classe URL con i seguenti costruttori e metodi:

- `URL(String s)` : crea l'URL relativa alla stringa `s` ; il costruttore deve sollevare una `IllegalArgumentException` se l'URL non ha il formato richiesto.
- `String prot()` : restituisce il protocollo.
- `String host()` : restituisce lo host.
- `String path()` : restituisce il path.
- `String query()` : restituisce la query se c'è, oppure `null` .
- `String fragm()` : restituisce il frammento se c'è, oppure `null` .
- `boolean sameHost(URL u)` : restituisce true se e solos e questa URL ha lo stesso host di quella passata come argomento.
- `URL absolute(String path)` : restituisce una URL che ha lo stesso protocollo e host di quella su cui è invocato, e ha il path uguale a quello passato come argomento (e nessuna query né frammento).

Classe PaginaWeb

Uno pagina web rappresenta il contenuto di una pagina web e il suo indirizzo. Per gli scopi di questo esame, il contenuto di una pagina è dato da una stringa che può contenere una o più sottostringhe della forma

```
<a href="URL">
```

dove `URL` è un'URL (che supponiamo non contenga virgolette al suo interno). Le sottostringhe indicate si chiamano *link* della pagina.

Scrivete i seguenti costruttori e metodi:

- `PaginaWeb(URL u, String p)` : costruisce la pagina web il cui indirizzo è `u` e il cui contenuto è `p` .
- `lista_di_URL link()` : restituisce i link della pagina. Potete decidere se deve restituire un array di URL o un `ArrayList<URL>` .
- `int locali()` : restituisce il numero di link locali contenuti in questa pagina; un link è detto locale se ha lo stesso host dell'URL della pagina.
- `int remoti()` : restituisce il numero di link remoti (cioè non locali) contenuti in questa pagina.

Classe Web

Il web rappresenta una collezione di pagine web. Idealmente tutte le pagine web del mondo. Ha i seguenti costruttori e metodi:

- `Web()` : costruisce il web vuoto (senza pagine).
- `void add(PaginaWeb p)` : aggiunge al web la pagina web `p` ; se per caso esiste già una pagina con lo stesso indirizzo di `p` , la sostituisce.
- `int n()` : restituisce il numero di pagine web esistenti.
- `PaginaWeb pagina(URL u)` : restituisce la pagina associata nel web all'indirizzo `u` , se esiste, oppure `null` .
- `boolean raggiunge(URL u1, URL u2, int passi)` : questo metodo determina se è possibile andare dalla pagina di indirizzo `u1` alla pagina di indirizzo `u2` in al più `passi` click. Si tratta di un metodo ricorsivo che funziona come segue:
  - se `passi=0` restituisce true se e solo se `u1` e `u2` sono uguali;
  - altrimenti, guarda la pagina associata all'URL `u1` :
    - se è `null` , restituisce false
    - altrimenti esamina tutti i link in essa contenuti, e per ognuno di essi (diciamo `x`) guarda se `raggiunge(x,u2,passi-1)` è vero; se sì, smette e restituisce true; altrimenti (se è falso per tutti), smette e restituisce false.
- `boolean raggiunge(String host1, String host2, int passi)` : restituisce true se e solo se esiste un'URL con host `host1` e un URL con host `host2` tali che si possa andare dalla prima alla seconda al più nel numero indicato di passi.