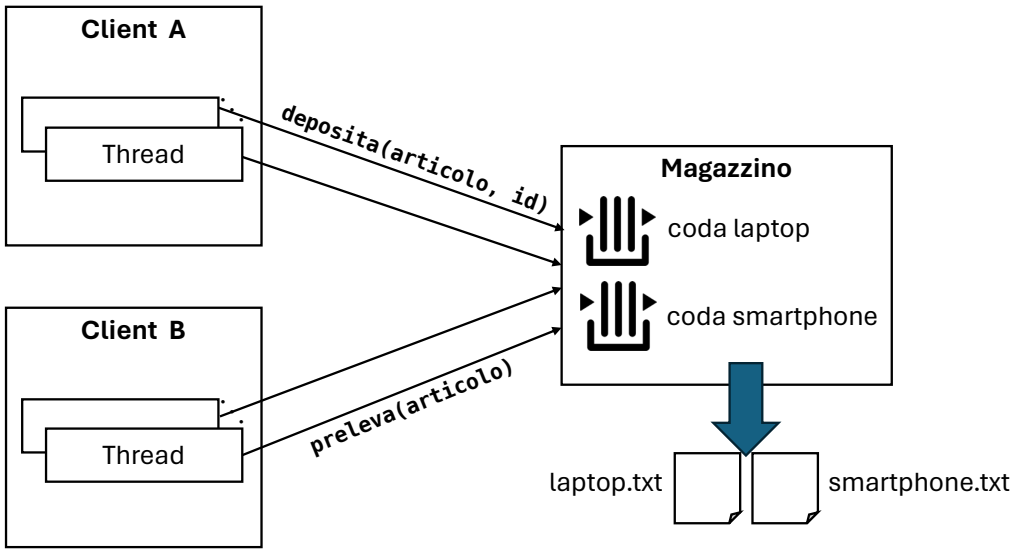
Università degli Studi di Napoli Federico II
Advanced Computer Programming
Esercitazione 01

Il candidato realizzi un'applicazione **Python multithread** per la gestione di un magazzino. Il sistema è composto da 3 tipologie di entità, come illustrato in figura:



1. **Magazzino** offre i servizi specificati dall'interfaccia `IMagazzino`. Il primo, **`deposita(articolo, id)`**, consente il deposito di un articolo. L'operazione di deposito è caratterizzata dal tipo di **articolo** (ossia "laptop" oppure "smartphone") e un **id** dell'articolo (selezionato a caso tra 1 e 100).
   Il Magazzino possiede 2 code a gestione circolare. Ciascun id è memorizzato nella coda specificata dal parametro **articolo.** Ogni coda è gestita secondo il problema produttore-consumatore. Se una coda è piena, la richiesta **deposita** è messa in attesa. Entrambe le code hanno dimensione pari 5.
   Il servizio **`preleva(articolo)`** restituisce l'id prelevato dalla coda specificata dal parametro articolo. Se la coda è vuota, l'operazione di prelievo è messa in attesa.
   Il Magazzino salva su file l'id dell'articolo prelevato.

2. **Client A**: genera 5 thread, ognuno dei quali, allo scadere di un tempo di *t* secondi (con t scelto a caso tra 2 e 4), effettua una richiesta **deposita**, con **articolo** ed **id** scelti a caso. Ogni thread effettua 3 richieste.

3. **Client B**: genera 5 thread, ognuno dei quali, allo scadere di un tempo di *t* secondi (con t scelto a caso tra 2 e 4), effettua una richiesta **preleva**, con **articolo** scelto a caso. Ogni thread effettua 3 richieste.

I Client possono essere implementati anche attraverso la stessa applicazione Python. Si utilizzi un parametro da terminale per discriminare il tipo di richiesta (cioè *deposita* o *preleva*).

Il candidato implementi il sistema utilizzando le socket ed il pattern proxy/skeleton.