COSA DOVREBBE ESSERE SOCRATE (v. Utopia)

Socrate è un programma che risolve problemi attraverso un sistema “enti/dati/incognite/proprietà/metodi”. Il programma è diviso in tre fasi.

**FASE DI INPUT**

In questa fase, attraverso l’inserimento di comandi, sotto forma di singoli caratteri, possono essere svolte varie azioni:

* Dichiarazione di un nuovo ente (‘e’): viene creato un “ente” con il nome desiderato.
* Selezionamento di un ente (‘s’): viene selezionato un ente già esistente.
* Dichiarazione di un nuovo dato (‘d’): viene aggiunto un dato, con nome e valore scelti dall’utente, all’ente selezionato.
* Dichiarazione di una nuova incognita (‘i’): analogo a sopra.
* Dichiarazione di una nuova proprietà (‘p’): analogo a sopra.
* Dichiarazione di un nuovo metodo (‘m’): viene aggiunto un nuovo metodo al file “metodi.txt”. Questa azione non dovrebbe essere fatta direttamente dal menu, ma è comunque possibile.
* Visualizzazione (‘v’): vengono visualizzate le opzioni attuali, l’ente selezionato e una lista di tutti gli enti esistenti con i loro rispettivi campi.
* Risoluzione (‘r’): viene risolto il problema con i dati che si hanno. Il programma poi viene terminato.
* Opzioni (vari caratteri): possono essere cambiati vari valori, che contribuiranno al modo in cui il programma risolverà il problema.

**FASE DI RISOLUZIONE**

Questa fase viene attivata dal comando “risolvi” nella fase di input. Per risolvere il problema, il programma non fa altro che sostituire a ogni stringa di procedimento che trova nel file metodi.txt prima le proprietà, e poi i dati, di ogni ente esistente, con lo scopo di trovare una “stringa risolvibile”, che risulterà nell’acquisizione di un risultato, che sarà aggiunto ai dati dell’ente interessato da esso. Il dato viene aggiunto in caso che possa essere riutilizzato nel prossimo svolgimento del “ciclo di risoluzione”, che viene infatti svolto nuovamente ogni volta che ne viene trovato uno nuovo [di dato]. Quando, durante lo svolgimento del ciclo, non viene ricavato nessun dato, la fase finisce. L’utente viene avvisato sia quando viene trovata una stringa risolvibile (per scopi di completezza dell’output), sia quando ne viene trovata una impossibile, poiché il ritrovamento di una di esse potrebbe risultare da dei dati o dei metodi incorretti (ES: diametro=10, raggio=2).

**FASE DI OUTPUT**

In questa fase vengono restituiti tutti i valori, trovati nella fase di risoluzione, che sono stati associati a un dato con lo stesso nome di un incognita presente su un ente.

**FASE DI APPRENDIMENTO**

Se tutte le incognite non sono state trovate, il programma chiederà all’utente di aggiungere dei metodi che lo aiutino a risolvere delle stringhe di procedimento a cui non sono stati sostituiti tutti i valori.

ESEMPIO (v. Corrente/Utopia)

**FASE DI INPUT**

* Il programma parte.
* Viene inserito un ente dal nome cerchio.
* Viene selezionato l’ente cerchio.
* Viene aggiunto un dato di nome diametro.
* Viene aggiunta un’incognita di nome area.
* Viene usato il comando di visualizzazione:
  + (opzioni)
  + Ente selezionato: cerchio
  + Enti esistenti:
  + -)cerchio
    - Dati associati:
    - -)diametro: 10
    - -)raggio: 2
    - Proprietà associate: nessuna
    - Incognite associate:
    - -)area
* Viene usato il comando di risoluzione.

**FASE DI RISOLUZIONE**

* Parte il primo ciclo di risoluzione.
* Viene trovato il metodo “Area Cerchio” (str proc: cerchio.area=cerchio.raggio\*cerchio.raggio\*3.14), dopo la fase di sostituzione si scopre che il metodo è risolvibile. La risoluzione (di nome “area” e di valore 2\*2\*3.14) viene aggiunta come nuovo dato all’ente cerchio. In piu, viene trovata una incognita, appartenente allo stesso ente, con lo stesso nome della risoluzione: l’incognita viene quindi cancellata, e la risoluzione aggiunta a un’array a parte, per essere poi stampata come output all’utente.
* Viene trovato il metodo “Rapporto Diametro-Raggio” (str proc: cerchio.diametro=2\*cerchio.raggio)