**CloudInterface**

*PopStudent* : ritorna l’ArrayList di Student che sono sulla Cloud e svuota la stessa Cloud

*PullStudent* : prende degli Student e li mette sulla Cloud

*isEmpty* : ritorna true se la Cloud è vuota, altrimenti falso

**DashboardInterface**

*addStudent* : aggiunge gli Student alla Hall

*setStudent* (Student, Island) : sposta uno Student su un’Island

*moveTowerToIsland* : sposta la Tower sull’Island

*moveTowerToDashboard* : riprende la Tower dall’Island

*setStudent* (Student) : sposta uno Student nella Hall

*checkInfluence* : ritorna vero se il Gamer ha la maggiore influenza su quel Color

**AssistantCardDeck**

*showOption* : ritorna le possibili AssistantCard da giocare

*selectedOption*(AssistantCard) : riceve l’AssistantCard scelta dal Gamer, la elimina dall’AssistantCardDeck e aggiorna currentSelection

**IslandInterface**

*refreshIslandState* : calcola l’influenza di ciascun Gamer sull’Island in cui è presente MotherNature (chiamata da Game)

*addStudent* : aggiunge uno Student all’Island

*mergeIsland*(Island) : unisce l’Island chiamante con l‘Island passata aggiornando il numero di Tower e il numero di Student

**GamerInterface**

*selectCloud*(Cloud) : prende una Cloud che il Gamer ha selezionato; su di essa si chiama il metodo *PopStudents*, quindi si chiama il metodo *addStudents* di *DashboardInterface*

**GameInterface**

*fillClouds* : per ogni Cloud viene riempita con nuovi Student randomici

*islandChecker*(Island) : calcola l’influenza di ogni Gamer sull’Island in cui si trova MotherNature e muove le Tower di conseguenza

*refreshMap*(Island) : controlla se le Island adiacenti hanno Tower dello stesso Color: se sì, allora chiama il metodo *mergeIsland* passando l’Island adiacente considerata

*getPossibleIsland* : riceve il currentPlayer e chiama currentPlayer.*getAssistantCardDeck*.*getCurrentSelection*.*getMovement* e ritorna di conseguenza le prime n Island che il Gamer può selezionare come possibile meta di MotherNature

*changeProfessorOwner*(Color, (current)Gamer) : per ogni Gamer diverso da currentGamer controlla se nella Hall di quel Color ha più Student di tutti. In tal caso cambia l’owner del Professor