***Contratti per lo UC “Gestire gli Eventi”***

* Pre-condizioni generali: l’attore deve essere identificato con un’istanza *org* di organizzatore

1. **VerificaStatoTurni()**

Pre-condizioni: -

Post-condizioni: -

1. **creazioneEvento(dataInizio: testo, dataFine: testo, nomeEvento: testo, luogo: testo,  
   numeroPartecipanti: intero, cliente: testo)**

Pre-condizioni:

* la data specificata non è precedente a quella in cui sta creando l’evento
* l’organizzatore gestisce l’evento

Post-condizioni:

* viene creata un istanza evento *evento*
* evento.stato = “in programma”
* evento.nome = nomeEvento
* evento.luogo = luogo
* evento.dataInizio = dataInizio
* evento.dataFine = dataFine
* evento.numeroPartecipanti = numeroPartecipanti
* evento.cliente = cliente

**2.a** **eliminazioneEvento(Evento: evento)**

Pre-condizioni:

* l’evento non è in stato terminato o in corso
* l’organizzatore gestisce l’evento

Post-condizioni:

* evento.stato = eliminato
* viene eliminata l’istanza evento specificata
* tutte le istanze servizio *s* associate a evento vengono eliminate
* tutte le istanze di incaricodiservizio *incServizio* relative a servizio vengono eliminate
* tutte le istanze di compito *c* associate al foglioripetilogativo *f di* servizio *s* vengono eliminate
* il personale e i cuochi tornano disponibili in quei turni

**2.b** **annullamentoEvento(Evento: evento)**

Pre-condizioni:

* l’evento non è in stato terminato

Post-condizioni:

* evento.stato = annullato
* viene eliminata l’istanza evento specificata
* tutte le istanze servizio s associate a evento vengono eliminate
* tutte le istanze di incaricodiservizio *incServizio* relative a servizio vengono eliminate
* tutte le istanze di compito c associate a servizio vengono eliminate
* il personale e lo chef tornano disponibili in quei turni

**2.c** **TerminazioneEvento(Evento: evento)**

Pre-condizioni:

* l’evento non è in stato terminato/annullato/eliminato

Post-condizioni:

* evento.stato = terminato

**2.c** **aperturaEvento(Evento: evento)**

Pre-condizioni:

* l’organizzatore gestisce l’evento

Post-condizioni: -

**2.c** **modificaEvento(Evento: evento, dataInizio?: testo, dataFine?: testo, nomeEvento?: testo,**

**luogo?: testo, numeroPartecipanti?: intero, Cliente?: testo)**

Pre-condizioni:

* [se inserita la data di inizio] allora evento.stato != in esecuzione
* [se inserita data di inizio] allora dataInizio > evento.dataInizio
* [se inserita dataFine] allora dataFine > evento.dataInizio
* [se inserito un numero di partecipanti] se evento.stato == confermato, allora il numero di partecipanti nuovi sono deve sforare del 30% rispetto al numero di partenza, altrimenti mora.

Post-condizioni:

* [se viene inserita la dataInizio] evento.dataInizio = dataInizio
* [se viene inserita la dataFine] evento. dataFine = durata
* [se viene inserita il nomeEvento] evento. nomeEvento = nomeEvento
* [se viene inserita il luogo] evento. luogo = luogo
* [se viene inserita il numeroPartecipanti] evento.numeroPartecipanti = numeroPartecipanti
* [se viene inserita il cliente] evento. cliente = cliente

1. **aggiungiServizio(nome: testo ,orarioInizio: orario, orarioFine: orario, qtyPersonale: qtyPersonale, dettagli: testo)**

Pre-condizioni:

* È in corso la visualizzazione dell’evento
* L’orario di inizio deve essere maggiore di quella dell’evento
* L’orario di inizio deve essere maggiore di quella attuale

Post-condizioni:

* viene creata un istanza di servizio s
* *s* forma evento evento *evento*
* s.orario = orario
* s.qtyPersonale = qtyPersonale
* s.dettaglio = dettaglio
* s.stato = programmato
* il servizio s forma l’evento ev

**3.a** **modificaServizio(Servizio: servizio, orarioInizio?: testo, orarioFine?: testo, qtyPersonale?: qtyPersonale, dettaglio?: testo)**

Pre-condizioni:

* è in corso la visualizzazione di un evento *evento* formato da servizi *s*
* il servizio non deve essere già segnato come terminato
* [se inserito l’orarioInizio] l’orario non deve essere già passato
* [se inserito l’orarioFine] l’orario non deve essere precedente a quello di inizio

Post-condizioni:

* [se inserito l’orario] s. orario = orario
* [se inserito il qtyPersonale] s. qtyPersonale = qtyPersonale
* [se inserito dettaglio] s.dettaglio = dettaglio

**3.b** **eliminaServizio(Servizio: servizio)**

Pre-condizioni:

* il servizio non deve essere già segnato come terminato
* è in corso la visualizzazione dell’evento

Post-condizioni:

* viene eliminata l’istanza S relativa a evento evento
* per ogni incarico di servizio da svolgere nel servizio viene tolta l’associazione
* ogni compito c relativo al turno t dall’istanza foglio riepilogativo fp del servizio s, vengono eliminati

1. **sceltaChef(Chef: chef)**

Pre-condizioni:

* è in corso la visualizzazione dell’evento

Post-condizioni:

* verrà assegnato uno chef *ch* al relativo evento *evento*

1. **approvazioneMenu(Servizio: servizio)**

Pre-condizioni:

* è in corso la visualizzazione di uno dei servizi di cui è formato l’evento
* lo chef ha già proposto un menu che è in uso per il servizio
* il servizio.stato non è terminato o annullato

Post-condizioni:

* l’evento.stato = “in corso”
* il menu *m* dievnta in uso per il servizio *s*

**5.a** **propostaModificaMenu(menu: menu, servizio: servizio, propostaModifica: propostaDiModifica)**

Pre-condizioni:

* è in corso la visualizzazione dell’evento suddiviso in servizi
* menu *m* è in uso nel servizio *s*
* il servizio non è in stato annullato o terminato

Post-condizioni:

* l’evento.stato = “in corso”
* proposta di modifica *pdm* modifica il Menu *m* in uso nel servizio *s*

1. **assegnamentoPersonale(Turno: turno, Servizio: servizio, Personale: personale, ruolo: ruolo)**

Pre-condizioni:

* il turno specificato non deve già essere terminato
* il turno specificato non deve essere saturo
* il servizio specificato non deve già essere terminato
* il personale deve essere disponibile in quel turno

Post-condizioni:

* verrà creata un istanza di incaricoDiServizio *incServizio*
* l’*incServizio* è da effettuare nel servizio *s*
* l’*incServizio* si effettua nel turno specificato
* l’*incServizio* è da effettuare nel servizio *s*
* il personale specificato svolge l’*incServizio*
* incServizio.ruolo = ruolo
* il personale svolge quell’*incServizio*

**6.a** **modificaAssegnamentoPersonale(Servizio: Servizio, Turno?: turno, Personale?: personale, ruolo?: ruolo)**

Pre-condizioni:

* [se specificato un turno] il turno specificato non deve già essere terminato
* [se specificato turno] il turno non deve essere saturo

Post-condizioni:

* verrà modificato l’incaricoDiServizio *incServizio* con i nuovi valori
* [se specificato il ruolo] incServizio.ruolo = ruolo
* [se specificato il turno] l’*incServizio* si effettua nel turno
* [se specificato il personale] l’*incServizio* viene svolto dal personale

**6.b** **eliminaAssegnamentoPersonale(IncaricoDiServizio: incaricoDiServizio)**

Pre-condizioni:

* È in corso la visualizzazione di un servizio
* Servizio.stato != terminato

Post-condizioni:

* È stata eliminata l’associazione svolge tra personale e incarico di servizio
* È stata eliminata l’associazione da effettuare tra incarico di servizio e servizio
* È stato eliminato l’incarico di servizio *incServizio*

1. **aggiungiNota (Servizio?: servizio, Evento?: evento, Descrizione: testo)**

pre-condizioni:

* è in corso la visualizzazione di un servizio oppure è in corso la visualizzazione di un evento

post-condizioni:

* verrà creata un istanza Nota N
* [se specificato il servizio] verrà creata un istanza Nota N relativa al Servizio
* [se specificato l’evento] verrà creata un istanza Nota N relativa al Servizio

**7.a** **modificaNota(Nota: Nota, Descrizione: testo)**

pre-condizioni: -

post-condizioni:

* verrà modificata l’istanza Nota N relativa al Servizio

**7.b** **eliminaNota(Servizio: servizio, Descrizione: testo)**

pre-condizioni: -

post-condizioni:

* verrà eliminata l’istanza Nota N relativa al Servizio