Travel to the Moon

1 Analisi dei Requisiti

1. Requisiti di Crociera
   - 1.1 codice: Stringa
   - 1.2 dataInizio: Data
   - 1.3 dataFine: Data
   - 1.4 associata alla Nave utilizzata
   - 1.5 associata a un solo Itinerario
   - 1.6 tipologie:{disj}
     - 1.6.1 Luna di Miele (Req.2)
     - 1.6.2 Famiglia
       - 1.6.2.1 bamb: Booleano
2. Requisiti di Luna di Miele
   - 2.1 divise in:{disj, compl}
     - 2.1.1 Tradizionali
       - 2.1.1.1 Destinazioni Romantiche $\geq$ Destinazioni Divertenti
     - 2.1.2 Alternative
       - 2.1.2.1 Destinazioni Romantiche $<$ Destinazioni Divertenti
3. Requisiti di Nave
   - 3.1 nome: Stringa
   - 3.2 comfort: 3..5
   - 3.3 capienza: Intero>0
4. Requisiti di Destinazione
   - 4.1 nome: Stringa
   - 4.2 continente: Continente
   - 4.3 associate a diversi Posti da vedere
   - 4.4 divise in:{}
     - 4.4.1 Romantiche
     - 4.4.2 Divertenti
5. Requisiti di Itenerario
   - 5.1 nome: Stringa
   - 5.2 associato ad una sequenza ordinata di Destinazioni sapendo:
     - 5.2.1 dataArrivo: OraTappa
     - 5.2.2 dataPartenza: OraTappa

6. Requisiti su Posto

   6.1 nome: Stringa

   6.2 descrizione: Stringa

   6.3 fasciaOraria: FasciaOra [1..*]

7. Requisiti di Cliente

   7.1 nome: Stringa

   7.2 cognome: Stringa

   7.3 età: Intero>0

   7.4 indirizzo: Indirizzo

   7.5 associato alle Prenotazioni che effettua

8. Requisiti di Prenotazioni

   8.1 istante: DataOra

   8.2 nPosti: Intero>0

   8.3 associato alla Crociera prenotata

2 Tipi di Dato

1. Continente: {Europa, Asia, Sud America, Nord America, Africa, Oceania, Antartide}
2. OraTappa: (giorno=Intero>0, orario=Ora)

   2.1 $<$ **(x:OraTappa, y:OraTappa): Booleano**

   - Pre-condizioni:
   - Post-condizioni:
     - result = (x.giorno $<$ y.giorno) $\vee$ (x.giorno = y.giorno $\wedge$ x.orario $<$ y.orario)

3. Giorno: {Lunedì, Martedì, Mercoledì, Giovedì, Venerdì, Sabato, Domenica}
4. FasciaOra: (giorno=Giorno, oraInizio=Ora, oraFine=Ora)
5. Indirizzo: (via=Stringa, civico=Intero>0)

3 Specifica delle Classi

1. **Specifica della classe Crociera:**

fine(): Data

- Pre-condizioni:
- Post-condizioni:
  - il result si calcola:
    - Sia i:Itinerario tale che (this,i):croc_itin
    - Sia x:OraTappa il valore "istante" del link (i,y):arr_itin
    - result = this.inizio + x.giorno

[**V.Crociera.date**]
Per ogni c:Crociera: c.inizio $\leq$ c.fine

2. **Specifica della classe Itinerario:**

[**V.Itinerario.arrivo_dopo_ultima_tappa**]
Per ogni i:Itinerario:

- Sia T l'insieme delle t:Tappa tali che (i,t):itin_tappa
- Sia x:OraTappa il valore "istante" del link (i,...):arr_itin

Per ogni t in T: t.dataPartenza < x

[**V.Itinerario.prima_tappa_dopo_partenza**]
Per ogni i:Itinerario:

- Sia T l'insieme delle t:Tappa tali che (i,t):itin_tappa e t.dataArrivo.giorno=1
- Sia x:OraTappa il valore "istante" del link (i,...):arr_itin

Per ogni t in T: t.dataArrivo.orario > x

[**V.Itinerario.arrivo_dopo_partenza_se_senza_tappe**]
Per ogni i:Itinerario:

- Sia T l'insieme delle t:Tappa tali che (i,t):itin_tappa
- Sia x:Ora il valore "istante" del link (i,...):part_itin
- Sia y:OraTappa il valore "istante" del link (i,...):arr_itin

Se $|T|$=0: x < y.orario

3. **Specifica della classe Tappa:**

nDest():Intero > 0

- Pre-condizioni:
- Post-condizioni:
  - il result si calcola:
    - Sia i:Itinerario tale che (this,i):itin_tappa
    - Sia T l'insieme delle t:Tappa tali che (i,t):itin_tappa e t.arrivo < this.arrivo
    - result = $|T|+1$

[**V.Tappa.date**]
Per ogni t:Tappa: t.arrivo < t.ripartenza