ESTENSIONE DI UN GIOCO D'AVVENTURA

Rusciano Marco

INDICE

0. INTRODUZIONE

0.1 - CENNI STORICI

I giochi di avventura nacquero negli anni settanta e prendono il nome dal titolo del primo programma di questo tipo, sviluppato da Crowther nel 1976 e poi perfezionato da Crowther e Woods nel 1977. Il titolo era “Colossal Cave Adventure”, o più semplicemente “Adventure”. In un gioco di questa tipologia è come se ci si trovasse ad essere i protagonisti di un film o di un romanzo, in cui però la trama non è già scritta, ma viene generata dinamicamente in base al nostro comportamento. Il gioco è caratterizzato da un ambiente (in genere costituito da uno o più luoghi), da una serie di oggetti fissi e trasportabili e da azioni eseguibili, il cui eventuale effetto può modificare lo stato del gioco, che verrà nel caso comunicato tramite stampa di un messaggio. Lo scopo di questa tipologia di giochi è raggiungere una determinata soluzione, che rappresenta la soluzione del gioco, risolvendo una serie di problemi e superando una serie di ostacoli. Il primo “Adventure” era puramente testuale, mentre in seguito si è passati all’inserimento di immagini statiche che accompagnano le descrizioni, fino ad arrivare ad “Adventure” con grafica 3D.

0.2 - TRACCIA

Il gioco su cui mi trovo a lavorare è “L’astronave condannata”, in cui il giocatore veste i panni del primo pilota. Il racconto inizia con il brusco risveglio da un sogno del protagonista principale, il quale capisce immediatamente che qualcosa non va a bordo dell’astronave che sta per esplodere. Il primo pilota dovrà quindi trovare un modo per evitare l’esplosione entro un certo limite di tempo. Nell’estensione del gioco richiesta si realizza una tipologia di oggetti contenitore trasportabili, e se ne inserisce un paio di istanze (zaino e valigia); di questa estensione vengono fatte analisi, progettazione e sviluppo. Nell’analisi viene specificata la definizione del problema e si determinano le operazioni che si dovranno effettuare. Nella progettazione vengono definite le linee essenziali della struttura del software in funzione di ciò che è stato scritto nell’analisi, cioè vengono definite le strutture dati che sono utilizzate (e le rispettive specifiche sintattiche e semantiche) e gli algoritmi utilizzati vengono rappresentati in pseudocodice. Infine nella fase di sviluppo viene implementato il modello concettuale e si ottiene quindi l’applicazione nel linguaggio di programmazione richiesto (C++).

1. ANALISI

Vista l'estensione del gioco richiesta, ho deciso di implementare due oggetti di tipo contenitore che siano trasportabili all'interno del gioco. I nomi di questi due oggetti sono "zaino" e "valigia". Chiaramente entrambi gli oggetti contenitore permettono di trasportare con se tutti gli oggetti che si trovano all'interno del gioco (ad eccezione della chiave, che può essere solo "portata con se"). E' inoltre possibile sia "trasportare con se" oggetti nello zaino e/o nella valigia, sia (allo stesso tempo) "indossarne" altri non trasportati.

Per distinguere il concetto di "oggetto zaino" da quello di "oggetto valigia" viene fatta una distinzione in termini di oggetti trasportabili al loro interno, quindi di "peso massimo" trasportabile. Viene quindi, a tal proposito, assegnato un valore "peso" ad ogni oggetto trasportabile. Consideriamo l'oggetto zaino di dimensione n, e l'oggetto valigia di dimensione m, con $n<m$, quindi con l'oggetto zaino che ci permette di trasportare oggetti di peso minore rispetto all'oggetto valigia. La posizione all'interno del gioco di questi due oggetti contenitore è fissa, cioè ad ogni nuova partita, l'oggetto zaino si troverà sempre nello stesso luogo (es. nella propria cabina) e sarà possibile decidere (come per qualsiasi altro oggetto all'interno del gioco) se utilizzarlo o meno. Stesso discorso vale per l'oggetto valigia, che chiaramente si trova in un luogo diverso dall'oggetto zaino, affinché il giocatore (riempito lo zaino, o qualora non abbia proprio preso lo zaino) possa o trasferire gli oggetti in un "contenitore" più capiente o trovare (nel corso del gioco) un'alternativa allo zaino. Non è possibile trasportare entrambi gli oggetti contenitore, quindi non è possibile trasportare con se sia lo zaino che la valigia, bisognerà nel corso dell'avventura sceglierne necessariamente uno dei due. La scelta del giocatore, tra zaino e valigia, non sarà solo una scelta "casuale", dettata dall'oggetto che verrà trovato prima all'interno del gioco, ma sarà anche una scelta ponderata, perché se la valigia è stata ideata come un oggetto che permette di trasportare oggetti al suo interno, è stata anche pensata come una loro collezione, della quale si ha immediatamente una panoramica precisa; mentre lo zaino (rifacendosi al concetto di zaino del mondo reale) è stato pensato come un contenitore di oggetti posti uno sopra l'altro, in cui è sempre possibile

utilizzare/prendere il primo oggetto che si troverà all'interno, cioè l'ultimo inserito: ciò significa che, per esplorarlo tutto, sarà necessario togliere tutti gli oggetti presenti a partire dalla cima, quindi ci vorranno tante mosse quanti sono gli oggetti meno uno, questo perché l'oggetto zaino non ha una "visione panoramica" di tutti gli oggetti presenti. Ho anche deciso di rivedere i comandi base quali "prendi", "lascia", "metti" e "indossa". Infatti il primo verrà utilizzato per prendere un oggetto ed infilarlo nello zaino o nella valigia(qualora quest'ultimi siano stati presi), ad eccezione dell'oggetto "manuale" che essendo molto importante per risolvere l'enigma finale potrà anche essere "portato con se" ed ad eccezione della chiave che potrà solo essere "portata con se". Il secondo invece verrà utilizzato per lasciare un oggetto, sia questo presente nello zaino o nella valigia (a condizione che quest'ultimi siano trasportati), sia questo presente nell'inventario (cioè trasportato con se). Infine gli ultimi due verranno utilizzati per "mettersi/indossare" gli oggetti (tuta, camice, casco) sia che essi si trovino nel luogo attualmente visitato, sia che essi si trovino nello zaino o nella valigia (a condizione che quest'ultimi siano traportati). Chiaramente a questo punto il comando "prendi" non potrà essere richiamato qualora non si sarà in possesso di uno zaino o di una valigia, sarà quindi presente un suggerimento che ci dirà di trovare prima un "oggetto contenitore" (sempre ad eccezione di chiave e manuale). Inoltre l'utilità di questi due oggetti contenitore all'interno del gioco si traduce anche nella possibilità, ad esempio, di poter "trasportare con se", in questi, sia la tuta che il camice, mentre nell' "Adventure" originale è possibile "indossarne" solo uno dei due. In definitiva ho deciso di fare una distinzione tra l'inventario già presente nel gioco e questi due oggetti contenitore. L'inventario presente nel gioco ci dirà ciò che "indossiamo" e ciò che "trasportiamo" (zaino, valigia, manuale, chiave); mentre l'oggetto contenitore ci dirà solo ciò che "trasportiamo" in esso, ma che non indossiamo ancora.

2. PROGETTAZIONE

Nella prima fase della progettazione individuiamo le strutture dati più adatte a rappresentare lo zaino e la valigia; mentre successivamente definiamo i metodi che consentono la loro gestione.

2.1 - PILA

Una pila è una sequenza di elementi di un certo tipo in cui è possibile aggiungere o togliere elementi solo da un estremo della sequenza (la "testa"). Può essere vista come un caso speciale di lista in cui l'ultimo elemento inserito è il primo ad essere rimosso (LIFO) e non è possibile accedere ad alcun elemento che non sia quello in testa.

2.1.1 - SPECIFICA SINTATTICA

Tipi:

- **Pila;**
- **Boolean;**
- **Tipoelem.**

Operatori:

- CREAPILA: () ⟶ **Pila**
- PILAVUOTA: (**Pila**) ⟶ **Boolean**
- LEGGIPILA: (**Pila**) ⟶ **Tipoelem**
- FUORIPILA: (**Pila**) ⟶ **Pila**
- INPILA: (**Tipoelem**, **Pila**) ⟶ **Pila**

2.1.2 - SPECIFICA SEMANTICA

Tipi:

- **Pila**: insieme delle sequenze P = a_1, a_2, …, a_n di elementi di tipo **Tipoelem** gestita con accesso LIFO;
- **Boolean**: insieme dei valori di verità.

Operatori:

- **CREAPILA** = P
  POST: P = Λ (La sequenza vuota)
- **PILAVUOTA** (P) = b
  POST: b = VERO se P $\equiv \Lambda$
  b = FALSO, altrimenti.

- **LEGGIPILA** (P) = a

  PRE: $P = a_1, a_2, \ldots, a_n$, $n \geq 1$

  POST: $a = a_1$

- **FUORIPILA** (P) = P'

  PRE: $P = a_1, a_2, \ldots, a_n$, $n \geq 1$

  POST: $P' = a_2, a_3, \ldots, a_n$, $n > 1$

  $P' = \Lambda$, se $n = 1$

- **INPILA** (a, P) = P'

  PRE: $P = a_1, a_2, \ldots, a_n$, $n \geq 0$

  POST: $P' = a, a_1, a_2, \ldots, a_n$

2.2 - INSIEME

Un insieme è una collezione/raccolta di elementi (componenti) di tipo omogeneo. A differenza delle liste gli elementi non sono caratterizzati da una posizione né possono apparire più di una volta. Solitamente in matematica sono rappresentati graficamente e possono essere definiti estensionalmente o intensionalmente (attraverso le proprietà che devono avere i componenti).
Il numero di elementi che compongono un insieme A è detto cardinalità di A e si indica con $|A|$. Se tale numero è finito, allora l'insieme ha cardinalità finita, altrimenti ha cardinalità infinita. Un insieme che non contiene alcun elemento è detto vuoto, e si indica con $\emptyset$. La relazione fondamentale per gli insiemi è l'appartenenza di un elemento x ad un insieme A, indicata formalmente con $x \in A$. Da questa si deriva l'inclusione tra due insiemi A e B: indicata con $A \subseteq B$, significa che ogni elemento x che appartiene ad A appartiene anche a B. Se $A \subseteq B$ e $B \subseteq A$, allora $A = B$, cioè A e B sono lo stesso insieme. Le operazioni principali tra due insiemi A e B sono l'unione, l'intersezione e la differenza, indicate, rispettivamente, con $A \cup B$, $A \cap B$ e $A - B$. L'unione denota l'insieme contenente tutti gli elementi di A e tutti gli elementi di B. L'intersezione denota l'insieme contenente tutti e soli gli elementi che appartengono sia ad A che a B. Infine la differenza denota l'insieme che contiene tutti e soli gli elementi che appartengono ad A ma non appartengono a B.

2.2.1 - SPECIFICA SINTATTICA

Tipi:

- **Insieme;**
- **Boolean;**
- **Tipoelem.**

Operatori:

- CREAINSIEME: () ⟶ **Insieme**
- INSIEMEVUOTO: (**Insieme**) ⟶ **Boolean**
- APPARTIENE: (**Tipoelem, Insieme**) ⟶ **Boolean**
- INSERISCI: (**Tipoelem, Insieme**) ⟶ **Insieme**
- CANCELLA: (**Tipoelem, Insieme**) ⟶ **Insieme**
- UNIONE: (**Insieme, Insieme**) ⟶ **Insieme**
- INTERSEZIONE: (**Insieme, Insieme**) ⟶ **Insieme**
- DIFFERENZA: (**Insieme, Insieme**) ⟶ **Insieme**

2.2.2 - SPECIFICA SEMANTICA

Tipi:

- **Insieme**: famiglia di insiemi costituita da elementi di tipo **Tipoelem**.
- **Boolean**: insieme dei valori di verità.

Operatori:

- **CREAINSIEME** = A
  POST: A = Ø
- **INSIEMEVUOTO** (A) = b
  POST: b = VERO se A = Ø
  b = FALSO, altrimenti.
- **APPARTIENE**(x, A) = b
  POST: b = VERO se x ∈ A
  b = FALSO, altrimenti.

- **INSERISCI** (x, A) = A’
  PRE: $x \notin A$
  POST: $A' = A \cup \{x\}$ (SE $x \in A$, $A \equiv A'$)
- **CANCELLA** (x, A) = A’
  PRE: $x \in A$
  POST: $A' = A - \{x\}$ (SE $x \notin A$, $A \equiv A'$)
- **UNIONE** (A, B) = C
  POST: $C = A \cup B$
- **INTERSEZIONE** (A, B) = C
  POST: $C = A \cap B$
- **DIFFERENZA** (A, B) = C
  POST: C = A - B

2.3 - VOCABOLARIO

All’interno del vocabolario del gioco ho aggiunto i seguenti termini:

- Inserisci (“aiuto”, 58, vocabolario)
- Inserisci (“help”, 58, vocabolario)
- Inserisci (“valigia”, 56, vocabolario)
- Inserisci (“zaino”, 57, vocabolario)

Quindi ogni termine nel vocabolario sarà costituito anche da un codice.

2.4 - OGGETTI

Ho aggiunto i seguenti oggetti a quelli inizialmente presenti:

- Inserisci(Oggetto(“una valigia”, 56, 5, 0)), oggetti);
- Inserisci(Oggetto(“uno zaino”, 57, 6, 0)), oggetti);

Quindi ogni oggetto sarà costituito dalla sua descrizione, dal suo codice, dal codice del luogo in cui è posto all’inizio del gioco ed infine dal suo peso. L’attributo peso è stato aggiunto per gestire al meglio ed in modo più realistico il peso massimo trasportabile nello zaino e nella valigia; gli oggetti con peso uguale a zero, sono oggetti il cui peso è stato trascurato all’interno del gioco, mentre ad ogni oggetto

trasportabile all'interno dello zaino e dalla valigia è stato attribuito un peso maggiore di zero. In particolare, abbiamo la seguente tabella dei pesi:

OGGETTO	PESO
Manuale	1
Tuta	2
Camice	3
Casco	3

Il peso all'interno del gioco è stato espresso in kg.

2.5 - AZIONI

Ho aggiunto le seguenti azioni:

- Inserisci(856, -2, azioni) // codice azione prendi valigia
- Inserisci(857, -2, azioni) // codice azione prendi zaino
- Inserisci(956, 3, azioni) // codice azione lascia valigia
- Inserisci(957, 3, azioni) // codice azione lascia zaino
- Inserisci(1056, -39, azioni) // codice azione guarda valigia
- Inserisci(1057, -40, azioni) // codice azione guarda zaino
- Inserisci(5600, 44, azioni) // codice azione inventario valigia
- Inserisci(5700, 42, azioni) // codice azione inventario zaino
- Inserisci(5800, 43, azioni) // codice azione aiuto/help

E modificato queste altre:

- Inserisci(2050, -2,azioni) //codice azione indossa/metti casco
- Inserisci(2051, -2, azioni) // codice azione indossa/metti tuta
- Inserisci(2052, -2, azioni) // codice azione indossa/metti camice

in

- Inserisci(2050, -41,azioni) // codice azione indossa/metti casco
- Inserisci(2051, -41, azioni) // codice azione indossa/metti tuta
- Inserisci(2052, -41, azioni) // codice azione indossa/metti camice

Quindi in azioni inserisco il codice del comando e il codice dell'azione da eseguire per quel comando. Il codice del comando è costituito da tre codici di 2 cifre ciascuno, per un totale di 6 cifre. Le prime due sono il codice del luogo (indicato con LL) in cui il comando è impartito, le due intermedie sono il codice del verbo (indicato con VV), che deve necessariamente essere presente e valido per indicare un comando, ed infine le ultime due sono il codice dell'oggetto (indicato con OO). Vediamo nel dettaglio le azioni inserite e modificate:

AZIONI	LL	VV	OO	CODICE
Prendi valigia	00	08	56	-2
Prendi zaino	00	08	57	-2
Lascia valigia	00	09	56	3
Lascia zaino	00	09	57	3
Guarda valigia	00	10	56	-39
Guarda zaino	00	10	57	-40
Inventario valigia	00	56	00	44
Inventario zaino	00	57	00	42
Aiuto/Help	00	58	00	43
Indossa/Metti casco	00	20	50	-41
Indossa/Metti tuta	00	20	51	-41
Indossa/Metti camice	00	20	52	-41

Per le azioni in cui si prende un oggetto ho usato il codice azione generale per "prendi oggetto" (il quale è stato comunque soggetto a modifiche), per le azioni in cui si lascia un oggetto ho usato il codice azione generale di "lascia oggetto" (soggetto a modifiche anch'esso), per le azioni in cui si guarda un oggetto ho creato un'azione apposita (azione 39 ed azione 40), per le azioni in cui si visiona l'inventario di un oggetto contenitore ho creato un'azione apposita (azione 42 ed azione 44), per le azioni in cui si visiona l'help del gioco ho creato un'azione apposita (azione 43), infine per distinguere l'azione "prendi oggetto" da "indossa oggetto", ho creato un'azione specifica in cui si indossa un oggetto (azione 41).

L'**azione prendi** consente di prendere un oggetto, se l'oggetto è un casco, una tuta o un camice, un suggerimento indicherà di trovare prima uno zaino o una valigia (oggetti contenitore); se l'oggetto contenitore starà già nell'inventario, si passerà al controllo dello spazio disponibile nello zaino o nella valigia, qualora non vi sia spazio

apparirà un suggerimento che ci indicherà di lasciare qualcosa di già presente in esso, altrimenti verrà inserito dentro, e ne avremo conferma con una stampa a video. Se l'oggetto sta già nell'inventario un avviso ce lo ricorderà e analogamente un avviso ci dirà se l'oggetto è già presente anche nello zaino o nella valigia. Infine se l'oggetto risulterà non prendibile (es. letto), una stampa ci dirà che è impossibile trasportarlo sia "addosso", sia "in uno zaino o in una valigia".

prendi() (gioco)

```
BEGIN
      IF (GET_LUOGO(GET_OGGETTO(og, oggetti)) = 0) THEN // controllo che
      l'oggetto che desidero prendere non si trovi già nell'inventario
            SCRIVI("- Già fatto.");
      ELSE IF(GET_LUOGO(GET_OGGETTO(og, oggetti)) = 12) THEN // controllo che
      l'oggetto che desidero non si trovi già nello zaino o nella valigia
            SCRIVI("- Sta già nello zaino o nella valigia.");
      ELSE IF (GET_LUOGO(GET_OGGETTO(og, oggetti)) < 0) THEN // controllo che
      l'oggetto che desidero sia prendibile
            SCRIVI("- Non è possibile.");
      ELSE IF (NOT PRENDI_SPECIFICHE()) THEN  // faccio alcuni controlli in
      prendi_specifiche
            IF(GET_LUOGO(GET_OGGETTO(25, oggetti)) = 0) THEN  // controllo se la
            valigia si trova nell'inventario
                  IF (og = 20 OR og = 23) THEN // controllo che l'oggetto sia il
                  secondo pilota o la chiave
                        SET_LUOGO(og, 0, oggetti);
                  ELSE
                        SET_LUOGO(og, 12, oggetti);
                  ENDIF
            ELSE IF(GET_LUOGO(GET_OGGETTO(26, oggetti)) = 0) THEN  // controllo
            se lo zaino si trova nell'inventario
                  IF (og = 20 OR og = 23) THEN // controllo che l'oggetto sia il
                  secondo pilota o la chiave
                        SET_LUOGO(og, 0, oggetti);
                  ELSE
                        SET_LUOGO(og, 12, oggetti);
                  ENDIF
            ELSE // se non ho né valigia, né zaino nell'inventario, e l'oggetto
            può essere preso(vedi. prendi_specifiche), lo metto nell'inventario
                  SET_LUOGO(og, 0, oggetti);
            ENDIF
            ENDIF
            IF (NOT PRESO_SPECIFICHE())THEN // faccio alcuni controlli in
            preso_specifiche
               SCRIVI("Fatto.");
            ENDIF
      ENDIF
      ENDIF
      ENDIF
      ENDIF
END
```

prendi_specifiche() (astro)

```
BEGIN
      problemi ← true;

      IF ((og = 4 || og = 11 || og = 22) AND (GET_LUOGO(GET_OGGETTO(25,
      oggetti))) != 0 AND (GET_LUOGO(GET_OGGETTO(26, oggetti))) != 0)) THEN //
      controllo che l'oggetto sia o la tuta, o il casco, o il camice e che sia
      valigia che zaino non si trovino nell'inventario
            SCRIVI("Trova prima uno zaino o una valigia.");
            SCRIVI("(Suggerimento: puoi comunque indossarlo.)");
      ELSE IF(og = 25 AND (GET_LUOGO(GET_OGGETTO(26, oggetti))) = 0) THEN //
      controllo che l'oggetto sia la valigia e che nell'inventario ci sia lo
      zaino
            SCRIVI("Lascia prima lo zaino.");  // linea 12
      ELSE IF(og = 26 AND (GET_LUOGO(GET_OGGETTO(25, oggetti))) = 0) THEN //
      controllo che l'oggetto sia lo zaino e che nell'inventario ci sia la
      valigia
            SCRIVI("Lascia prima la valigia.");  // linea 12
      ELSE // se non si verifica nessuna di queste ipotesi, imposto problemi a
      false
            problemi ← false;
      return (problemi);
      ENDIF
      ENDIF
      ENDIF

END
```

preso_specifiche() (astro)

```
BEGIN
      avvisato ← true;

      IF ((og = 4 || og = 11 || og = 22 || og = 24) AND
      ((GET_LUOGO(GET_OGGETTO(25, oggetti))) = 0)) THEN  // controllo che
      l'oggetto sia o la tuta, o il casco, o il camice, o il manuale e che la
      valigia stia nell'inventario
            IF((GET_PESO(GET_OGGETTO(og, oggetti))) > peso_MaxV AND og != 24)
            THEN  // controllo che il peso dell'oggetto sia maggiore allo spazio
            disponibile nella valigia e che l'oggetto sia diverso dal manuale
                  SET_LUOGO(og, luogo_attuale, oggetti);
                  SCRIVI("La valigia è troppo piena...");
                  SCRIVI("(Suggerimento: togli qualcosa dalla valigia.)");
            ELSE IF((GET_PESO(GET_OGGETTO(og, oggetti))) > peso_MaxV AND og=24)
            THEN  // controllo che il peso dell'oggetto sia maggiore allo spazio
            disponibile nella valigia e che l'oggetto sia uguale al manuale
                  SET_LUOGO(og, 0, oggetti);
                  SCRIVI("La valigia e' troppo piena...");
                  SCRIVI("Lo tengo in mano, e' troppo importante.");
                  SCRIVI("(Suggerimento: controlla nell'inventario)");
            ELSE// se non si verifica nessuna di queste ipotesi l'oggetto viene
            inserito in valigia
                  SCRIVI("Ora e' in valigia.");
                  INSERISCI((GET_CODICE(GET_OGGETTO(og, oggetti))), INSIEME);
                  n_oggettiV++;
```

```
                peso_MaxV = peso_MaxV - (GET_PESO(GET_OGGETTO(og, oggetti)));
            ENDIF
            ENDIF
        ELSE IF ((og = 4 || og = 11 || og = 22 || og = 24) AND
        (GET_LUOGO(GET_OGGETTO(26, oggetti))) = 0)) THEN  // controllo che
        l'oggetto sia o la tuta, o il casco, o il camice, o il manuale e che lo
        zaino stia nell'inventario
            IF((GET_PESO(GET_OGGETTO(og, oggetti))) > peso_MaxZ AND og != 24)
            THEN  // controllo che il peso dell'oggetto sia maggiore allo spazio
            disponibile nello zaino e che l'oggetto sia diverso dal manuale
                SET_LUOGO(og, luogo_attuale, oggetti);
                SCRIVI("Lo zaino e' troppo pieno...");
                SCRIVI("(Suggerimento: togli qualcosa dallo zaino.)");
            ELSE IF((GET_PESO(GET_OGGETTO(og, oggetti))) > peso_MaxZ AND og=24)
            THEN  // controllo che il peso dell'oggetto sia maggiore allo spazio
            disponibile nello zaino e che l'oggetto sia uguale al manuale
                SET_LUOGO(og, 0, oggetti);
                SCRIVI("Lo zaino e' troppo pieno...");
                SCRIVI("Lo tengo in mano, e' troppo importante.");
                SCRIVI("(Suggerimento: controlla nell'inventario)");
            ELSE // se non si verifica nessuna di queste ipotesi l'oggetto viene
            inserito nello zaino
                SCRIVI("Ora e' nello zaino.");
                INPILA((GET_CODICE(GET_OGGETTO(og, oggetti))), PILA);
                n_oggettiZ++;
                peso_MaxZ = peso_MaxZ - (GET_PESO(GET_OGGETTO(og, oggetti)));
            ENDIF
            ENDIF
        ELSE // se non si verifica nessuna di queste ipotesi, imposto avvisato a
        false
            avvisato ← false;
        return (avvisato);
        ENDIF
        ENDIF

END
```

L'**azione lascia** consente di lasciare un oggetto, se l'oggetto non si troverà nel luogo attuale, né nell'inventario, né in valigia, né nello zaino un avviso ci indicherà che l'oggetto non solo non è presente nel luogo in cui ci troviamo, ma anche che non lo trasportiamo; se invece l'oggetto pur non essendo né nell'inventario, né in valigia, né nello zaino è presente nel luogo attuale, un avviso ci indicherà che non lo abbiamo ancora preso. Invece se l'oggetto verrà "lasciato" all'esterno dell'astronave, una stampa a video ci dirà che l'oggetto ormai è andato perduto per sempre nello spazio. I casi cambiano invece quando abbiamo nell'inventario lo zaino o la valigia. Nel primo caso infatti controlliamo che l'oggetto sia veramente presente nello zaino e qualora lo sia lo lasciamo nel luogo attuale, questo a condizione che l'oggetto sia il primo "visibile" nello zaino, nel caso in cui, invece l'oggetto si trovi al di sotto del primo visibile, sarà necessario togliere tutti gli oggetti presenti prima di esso, infine se l'oggetto non sarà proprio presente nello zaino un avviso ci dirà che non si trova in esso. Nel secondo caso invece controlliamo che l'oggetto sia

veramente presente nella valigia e qualora lo sia lo lasciamo nel luogo attuale, ed a differenza dello zaino, avendo la valigia una visione globale di tutti gli oggetti, sarà sempre immediatamente possibile lasciare un oggetto in essa presente. Chiaramente se l'oggetto non risulta presente in valigia, un avviso ci dirà che non si trova nella valigia. Infine se verranno lasciati casco o tuta all'esterno dell'astronave, il protagonista del gioco morirà, essendo casco e tuta essenziali per muoversi all'esterno.

lascia() (gioco)

```
BEGIN

      IF (og = 0 AND GET_LUOGO(GET_OGGETTO(og, oggetti)) != 0 AND
      GET_LUOGO(GET_OGGETTO(og, oggetti)) != 12) THEN // controllo che l'oggetto
      si trovi in un altro luogo e che l'oggetto non si trovi né
      nell'inventario, né nella valigia, né nello zaino
            SCRIVI("- Non ce l'hai e qui non c'e'.");
      ELSE IF(og != 0 AND GET_LUOGO(GET_OGGETTO(og, oggetti)) != 0 AND
      GET_LUOGO(GET_OGGETTO(og, oggetti)) != 12) THEN // controllo che l'oggetto
      non si trovi in un altro luogo e che l'oggetto non si trovi né
      nell'inventario, né nella valigia, né nello zaino
            SCRIVI("- Non ce l'hai.");
      ELSE IF (og = 0 AND GET_LUOGO(GET_OGGETTO(og, oggetti))!= 0) THEN //
      controllo che l'oggetto si trovi in un altro luogo e che l'oggetto non si
      trovi nell'inventario
            SCRIVI("- Non ce l'hai.");
      ELSE IF (NOT LASCIA_SPECIFICHE())  // faccio alcuni controlli in
      lascia_specifiche
            IF(GET_LUOGO(GET_OGGETTO(26, oggetti)) = 0) THEN  // controllo che
            lo zaino si trovi nell'inventario
                  IF (GET_LUOGO(GET_OGGETTO(4, oggetti)) = 12 OR
                  GET_LUOGO(GET_OGGETTO(11, oggetti)) = 12 OR
                  GET_LUOGO(GET_OGGETTO(22, oggetti)) = 12 OR
                  GET_LUOGO(GET_OGGETTO(24, oggetti)) = 12) THEN { // controllo
                  che uno tra:tuta, casco, camice e manuale si trovi nello zaino
                        IF(og = 4 OR og = 11 OR og = 22 OR og = 24) THEN  //
                        controllo che l'oggetto sia proprio uno tra: tuta,
                        casco, camice e manuale
                              IF(GET_LUOGO(GET_OGGETTO(og, oggetti)) = 0) THEN
                              // controllo che l'oggetto si trovi
                              nell'inventario
                                    SET_LUOGO(og, luogo_attuale, oggetti);
                                    SCRIVI("Fatto.");
                              ELSE // controllo che l'oggetto si trovi nello
                              zaino
                                    c ← LEGGIPILA(PILA);
                                    d ← og;
                                    IF(og = GET_ZAINO2(c, oggetti)) THEN //
                                    controllo che l'oggetto sia il primo
                                    visibile nello zaino
                                          SET_LUOGO(og, luogo_attuale, oggetti);
                                          FUORIPILA(PILA);
                                          n_oggettiZ--;
                                          peso_MaxZ = peso_MaxZ +
```

```
                                    GET_PESO(GET_OGGETTO(og, oggetti));
                                    SCRIVI("Fatto.");
                              ELSE IF (APPARTIENE(GET_VALIGIA(d, oggetti),
                              INSIEME)) THEN  // controllo che l'oggetto
                              si trovi nella valigia
                                    SCRIVI("- Non ce l'hai addosso e non
                                    sta nello zaino..");
                                    SCRIVI(" (Suggerimento: controlla nella
                                    valigia)");
                              ELSE IF (NOT PILAVUOTA(PILA)) THEN  //
                              controllo che lo zaino non sia vuoto
                                    SCRIVI("Devi prima lasciare:");
                                    GET_ZAINO(c, oggetti);
                              ENDIF
                              ENDIF
                              ENDIF
                        ENDIF
                  ELSE
                        SET_LUOGO(og, luogo_attuale, oggetti);
                        SCRIVI("Fatto.");
                  ENDIF
            ELSE
                  IF(GET_LUOGO(GET_OGGETTO(og, oggetti)) != 0) THEN  //
                  controllo che l'oggetto non si trovi nell'inventario
                        SCRIVI("- Non ce l'hai.");
                  ELSE// controllo che l'oggetto si trovi nell'inventario
                        SET_LUOGO(og, luogo_attuale, oggetti);
                        SCRIVI("Fatto.");
                  ENDIF
            ENDIF
      ELSE IF(GET_LUOGO(GET_OGGETTO(25, oggetti)) = 0) THEN // controllo
      che la valigia si trovi nell'inventario
            IF (GET_LUOGO(GET_OGGETTO(4, oggetti)) = 12 OR
            GET_LUOGO(GET_OGGETTO(11, oggetti)) = 12 OR
            GET_LUOGO(GET_OGGETTO(22, oggetti)) = 12 OR
            GET_LUOGO(GET_OGGETTO(24, oggetti)) = 12) THEN // controllo
            che uno tra: tuta, casco, camice e manuale si trovi nella
            valigia
                  IF(og = 4 OR og = 11 OR og = 22 OR og = 24) THEN  //
                  controllo che l'oggetto sia proprio uno tra: tuta,
                  casco, camice e manuale
                        c ← og;
                        IF(GET_LUOGO(GET_OGGETTO(og, oggetti)) = 0) THEN
                        //controllo che l'oggetto si trovi nell'inventario
                              SET_LUOGO(og, luogo_attuale, oggetti);
                              SCRIVI("Fatto.");
                        ELSE IF(NOT APPARTIENE(GET_VALIGIA(c, oggetti),
                        INSIEME)) THEN// controllo che l'oggetto non si
                        trovi nella valigia
                              IF((GET_VALIGIA(c, oggetti)) =
                              (LEGGIPILA(PILA))) THEN // controllo che
                              l'oggetto sia il primo visibile dello zaino
                                    SCRIVI("- Non ce l'hai in valigia.");
                                    SCRIVI(" (Suggerimento: controlla nello
                                    zaino)");
                              ELSE IF((GET_VALIGIA(c, oggetti)) !=
                              (LEGGIPILA(PILA))) THEN // controllo che
                              l'oggetto non sia il primo visibile nello
                              zaino
```

```
                                    SCRIVI("- Non ce l'hai in valigia.");
                              ENDIF
                              ENDIF
                        ELSE IF(APPARTIENE(GET_VALIGIA(c, oggetti),
                        INSIEME)) THEN // controllo che l'oggetto si trovi
                        nella valigia
                              SET_LUOGO(og, luogo_attuale, oggetti);
                              CANCELLA(GET_VALIGIA(c, oggetti), INSIEME);
                              n_oggettiV--;
                              peso_MaxV = peso_MaxV +
                              GET_PESO(GET_OGGETTO(og, oggetti));
                              SCRIVI("Fatto.");
                        ELSE
                              SCRIVI("- Non ce l'hai in valigia.");
                        ENDIF
                        ENDIF
                        ENDIF
                  ELSE
                        SET_LUOGO(og, luogo_attuale, oggetti);
                        SCRIVI("Fatto.");
                  ENDIF
            ELSE
                  IF(GET_LUOGO(GET_OGGETTO(og, oggetti)) != 0) THEN  //
                  controllo che l'oggetto non si trovi nell'inventario
                        SCRIVI("- Non ce l'hai.");
                  ELSE
                        SET_LUOGO(og, luogo_attuale, oggetti);
                        SCRIVI("Fatto.");
                  ENDIF
            ENDIF
      ELSE
            IF(GET_LUOGO(GET_OGGETTO(og, oggetti)) = 0) THEN  // controllo
            che l'oggetto si trovi nell'inventario
                  SET_LUOGO(og, luogo_attuale, oggetti);
                  SCRIVI("Fatto.");
            ELSE
                  SCRIVI("- Non ce l'hai.");
            ENDIF
      ENDIF
      ENDIF
ELSE IF ((luogo_attuale = 7 OR luogo_attuale >= 9) AND
(GET_LUOGO(GET_OGGETTO(4, oggetti)) != 0 OR GET_LUOGO(GET_OGGETTO(11,
oggetti)) != 0)) THEN  // controllo che il luogo attuale sia o 7 o
qualsiasi luogo pari o superiore a 9 e che la tuta o il casco non si
trovino nell'inventario
      SCRIVI("\nAaaagh!!!");
      MORTO();
ENDIF
ENDIF
ENDIF
ENDIF
ENDIF
END
```

lascia_specifiche() (astro)

```
BEGIN
      lasciato ← true;
      IF (luogo_attuale >= 9 OR (luogo_attuale = 7 AND parete_stagna_aperta =1))
      THEN  //controllo che il luogo sia pari o superiore a 9 (prua astronave,
      esterno astronave, poppa astronave) o che il luogo sia il compartimento
      stagno e che la combinazione dei pulsanti(rosso-verde) sia 1, quindi tasto
      rosso premuto
            SET_LUOGO(og, -99, oggetti);
            SCRIVI("Si e' perso nello spazio.");
      ELSE
            lasciato ← false;
      ENDIF
      return (lasciato);

END
```

A seguire 3 funzioni utilizzate nell'**azione lascia**:

get_zaino() (oggetti)

```
BEGIN
      trovato ← false;
      i ← 1;
      WHILE (i <= fine_Oggetti AND NOT trovato) DO // scandisco tutti gli
      oggetti presenti nel vocabolario finché o finiscono gli oggetti o trovato
      viene impostato a true
            IF (GET_CODICE(oggetti[i]) = c) THEN // controllo che il codice
            dell'oggetto con indice "i" sia uguale al codice "c"
                  SCRIVI(GET_NOME(oggetti[i]));
                  trovato ← true;
            i++;
      ENDWHILE
END
```

get_zaino2() (oggetti)

```
BEGIN
      trovato ← false;
      i ← 1;
      i2 ← 0;
      WHILE (i <= fine_Oggetti AND NOT trovato) DO // scandisco tutti gli
      oggetti presenti nel vocabolario finché o finiscono gli oggetti o trovato
      viene impostato a true
            IF (GET_CODICE(oggetti[i]) = c) THEN // controllo che il codice
            dell'oggetto con indice "i" sia uguale al codice "c"
                  i2 ← i;
                  trovato ← true;
            ENDIF
            i++;
      ENDWHILE
      return(i2);

END
```

get_valigia() (oggetti)

```
BEGIN
      trovato ← false;
      i ← 1;
      i2 ← 0;
      WHILE (i <= fine_Oggetti AND NOT trovato) DO // scandisco tutti gli
      oggetti presenti nel vocabolario finché o finiscono gli oggetti o trovato
      viene impostato a true
            IF (i = c) THEN //controllo che l'indice "i" sia uguale al codice c
                  i2 ← GET_CODICE(oggetti[i]);
                  trovato ← true;
            ENDIF
            i++;
      ENDWHILE
      return(i2);

END
```

L'**azione 39 (guarda valigia)** permette di ottenere informazioni sull'oggetto valigia.

azione_39() (astro)

```
BEGIN
      SCRIVI("E' la valigia del secondo pilota.");
      SCRIVI("Sembra molto capiente.");

END
```

L'**azione 40 (guarda zaino)** permette di ottenere informazioni sull'oggetto zaino.

azione_40() (astro)

```
BEGIN
      SCRIVI("E' il tuo zaino, puo' essere utile per trasportare oggetti.");
END
```

L'**azione 44 (inventario valigia)** permette di visionare il contenuto della valigia, ottenendo informazioni quali:

- nome oggetti contenuti;
- numero di oggetti trasportati;
- spazio disponibile ed occupato in kg.

azione_44() (astro)

```
BEGIN
      IF(GET_LUOGO(GET_OGGETTO(25, oggetti)) = 0) THEN // controllo che
      l'oggetto valigia si trovi nell'inventario
```

```
            IF(NOT INSIEMEVUOTO(INSIEME)) THEN // controllo che l'oggetto
            valigia non sia vuoto
                  SCRIVI("Inventario Valigia");
                  SCRIVI("Vedo:");
                  IF(APPARTIENE(50, INSIEME)) THEN // controllo che il casco sia
                  presente nella valigia
                        GET_ZAINO(50, oggetti);
                  ENDIF
                  IF(APPARTIENE(51, INSIEME)) THEN // controllo che la tua sia
                  presente nella valigia
                        GET_ZAINO (51, oggetti);
                  ENDIF
                  IF(APPARTIENE(52, INSIEME)) THEN // controllo che il camice
                  sia presente nella valigia
                        GET_ZAINO (52, oggetti);
                  ENDIF
                  IF(APPARTIENE(55, INSIEME)) THEN // controllo che il manuale
                  sia presente nella valigia
                        GET_ZAINO(55, oggetti);
                  ENDIF
                  SCRIVI("Totale Oggetti nella Valigia:" + n_oggettiV);
                  SCRIVI("Spazio disponibile: " + peso_MaxV + " su 8 kg.");
            ELSE
                  SCRIVI("E' vuota.");
            ENDIF
      ELSE
            SCRIVI("Non ce l'hai.");
      ENDIF
END
```

L'**azione 42 (inventario zaino)** permette di visionare il contenuto della valigia, ottenendo informazioni quali:

- nome primo oggetto contenuto;
- numero di oggetti trasportati;
- spazio disponibile ed occupato in kg.

azione_42() (astro)

```
BEGIN
      IF(GET_LUOGO(GET_OGGETTO(26, oggetti)) = 0) THEN // controllo che
      l'oggetto zaino si trovi nell'inventario
            IF (NOT PILAVUOTA(PILA)) THEN // controllo che l'oggetto zaino non
            sia vuoto
                  SCRIVI("Inventario Zaino");
                  SCRIVI("Vedo in cima: ");
                  c ← LEGGIPILA(PILA);
                  GET_ZAINO(c, oggetti);
                  SCRIVI("Totale Oggetti nello Zaino:" + n_oggettiZ );
                  SCRIVI("Spazio disponibile: "+ peso_MaxZ + " su 5 kg.");
            ELSE
                  SCRIVI("E' vuoto.");
            ENDIF
      ELSE
```

```
            SCRIVI("Non lo hai.");
      ENDIF
END
```

L'**azione 43 (aiuto/help)** permette di ottenere informazioni sui comandi di gioco.

azione_43() (astro)

```
BEGIN
      SCRIVI("COMANDI DI GIOCO:");
      SCRIVI("Direzioni: ");
      SCRIVI("- n/nord: per muoverti in avanti;");
      SCRIVI("- s/sud: per muoverti indietro;");
      SCRIVI("- e/est: per muoverti a destra;");
      SCRIVI("- w/o/ovest: per muoverti a sinistra;");
      SCRIVI("- a/alto/sali: per salire ad un piano superiore;");
      SCRIVI("- b/basso/scendi: per scendere ad un piano inferiore;");
      SCRIVI("Azioni: ");
      SCRIVI("- prendi: per trasportare un oggetto in mano o con se
      (zaino/valigia);");
      SCRIVI("- indossa/metti: per indossare un oggetto(es. casco);");
      SCRIVI("- guarda: per guardare ed ottenere informazioni su un oggetto
      (es.tuta);");
      SCRIVI("- lascia/togli/leva: per lasciare o togliersi gli oggetti
      trasportati;");
      SCRIVI("- apri: per aprire un oggetto fisso(es. armadietto);");
      SCRIVI("- leggi: per leggere una scritta(es.cartello);");
      SCRIVI("- spingi/tira: per spingere o tirare un oggetto fisso(es.leva);");
      SCRIVI("- premi/schiaccia: per premere un oggetto fisso(es.pulsante);");
      SCRIVI("- inventario/cosa: per accedere all'inventario degli oggetti
      trasportati;");
      SCRIVI("- zaino: per accedere agli oggetti trasportati nello zaino;");
      SCRIVI("- valigia: per accedere agli oggetti trasportati nella valigia;");
      SCRIVI("- save/load: per salvare o caricare la partita;");
END
```

L'**azione indossa** consente di indossare un oggetto ("metterlo nell'inventario"). Se l'oggetto risulterà già nell'inventario, allora un avviso ci informerà che l'azione è stata già eseguita in precedenza, invece se l'oggetto risulterà non prendibile (es. letto), una stampa a video ci dirà che non è possibile indossarlo. Inoltre verranno fatti due ulteriori controlli riguardanti gli oggetti: tuta e camice; infatti non è possibile indossarli entrambi. Allora si verificherà che all'atto dell'indossare uno dei due, l'altro non sia già indossato. I casi cambiano invece quando abbiamo nell'inventario lo zaino o la valigia. Nel primo caso infatti verifichiamo che l'oggetto che si vuole indossare stia effettivamente nello zaino, e qualora si trovi nella prima posizione verrà indossato, altrimenti un avviso ci suggerirà di lasciare prima tutti gli oggetti che in ordine sono stati inseriti prima dell'oggetto desiderato. Inoltre

l'azione indossa verrà eseguita correttamente anche nel caso in cui, nonostante lo zaino stia nell'inventario, l'oggetto si trovi semplicemente nel luogo attuale. Analogamente nel secondo caso verifichiamo che l'oggetto che si vuole indossare stia effettivamente nella valigia, se ciò è vero, allora l'oggetto verrà immediatamente indossato, altrimenti un messaggio a video ci suggerirà che l'oggetto o si trova nello zaino oppure non è proprio trasportato. Come nel caso dello zaino, anche in questo, se l'oggetto valigia si troverà nell'inventario e l'oggetto desiderato si troverà nel luogo attuale, sarà comunque possibile indossarlo, e quindi portare a termine correttamente l'azione.

azione_41() (astro)

```
BEGIN
      IF (GET_LUOGO(GET_OGGETTO(og, oggetti)) = 0) THEN // controllo che
      l'oggetto si trovi nell'inventario
            SCRIVI("- Gia' fatto.");
      ELSE IF(GET_LUOGO(GET_OGGETTO(og, oggetti)) < 0) THEN // controllo che
      l'oggetto non sia indossabile
            SCRIVI("- Non e' possibile.");
      ELSE IF(NOT INDOSSA_SPECIFICHE())  // faccio alcuni controlli in
      indossa_specifiche
            IF(GET_LUOGO(GET_OGGETTO(26, oggetti)) = 0) THEN  // controllo che
            lo zaino sia presente nell'inventario
                  IF(GET_LUOGO(GET_OGGETTO(4, oggetti)) = 12 OR
                  GET_LUOGO(GET_OGGETTO(11, oggetti)) = 12 OR
                  GET_LUOGO(GET_OGGETTO(22, oggetti)) = 12) THEN // controllo
                  che almeno uno tra: tuta, casco e camice sia presente nello
                  zaino
                        IF(og = 4 OR og = 11 OR og = 22) THEN  // controllo che
                        l'oggetto in questione sia proprio uno tra: tuta, casco
                        e camice
                              c ← LEGGIPILA(PILA);
                              IF(og = GET_ZAINO2(c, oggetti)) THEN // controllo
                              che l'oggetto sia proprio il primo oggetto nello
                              zaino
                                    SET_LUOGO(og, 0, oggetti);
                                    FUORIPILA(PILA);
                                    n_oggettiZ--;
                                    peso_MaxZ = peso_MaxZ +
                                    GET_PESO(GET_OGGETTO(og, oggetti));
                              ELSE IF(GET_LUOGO(GET_OGGETTO(og, oggetti)) =
                              luogo_attuale) THEN // controllo che l'oggetto si
                              trovi nel luogo attuale
                                    SET_LUOGO(og, 0, oggetti);
                              ELSE
                                    SCRIVI("Devi prima lasciare: ");
                                    GET_ZAINO(c, oggetti);
                              ENDIF
```

```
                        ENDIF
                  ELSE
                        SET_LUOGO(og, 0, oggetti);
                        SCRIVI("Fatto.");
                  ENDIF
            ELSE
                  SET_LUOGO(og, 0, oggetti);
            ENDIF
      ELSE IF(GET_LUOGO(GET_OGGETTO(25, oggetti)) = 0) THEN // controllo
      che la valigia si trovi nell'inventario
            IF(GET_LUOGO(GET_OGGETTO(4, oggetti)) = 12 OR
            GET_LUOGO(GET_OGGETTO(11, oggetti)) = 12 OR
            GET_LUOGO(GET_OGGETTO(22, oggetti)) = 12) THEN // controllo
            che almeno uno tra: tuta, casco e camice si trovi nella
            valigia
                  IF(og = 4 OR og = 11 OR og = 22) THEN  // controllo che
                  l'oggetto sia proprio uno tra: tuta, casco e camice
                        c ← og;
                        IF(APPARTIENE(GET_VALIGIA(c, oggetti), INSIEME))
                        THEN// controllo che l'oggetto sia presente nella
                        valigia
                              SET_LUOGO(og, 0, oggetti);
                              CANCELLA(GET_VALIGIA(c, oggetti));
                              n_oggettiV--;
                              peso_MaxV = peso_MaxV +
                              GET_PESO(GET_OGGETTO(og, oggetti));
                        ELSE IF(NOT APPARTIENE(GET_VALIGIA(c, oggetti),
                        INSIEME)) THEN //controllo che l'oggetto non sia
                        presente nella valigia
                              SCRIVI("- Non ce l'hai in valigia.");
                        ELSE IF(GET_LUOGO(GET_OGGETTO(og, oggetti)) =
                        luogo_attuale) THEN //controllo che l'oggetto si
                        trovi nel luogo attuale
                              SET_LUOGO(og, 0, oggetti);
                        ENDIF
                        ENDIF
                        ENDIF
                  ELSE
                        SET_LUOGO(og, 0, oggetti);
                        SCRIVI("Fatto.");
                  ENDIF
            ELSE
                  SET_LUOGO(og, 0, oggetti);
            ENDIF
      ELSE
            IF(og != 0 AND GET_LUOGO(GET_OGGETTO(og, oggetti)) != 12)THEN
            // controllo che l'oggetto non si trovi in un altro luogo e
            che l'oggetto non si trovi nello zaino o nella valigia
                  SET_LUOGO(og, 0, oggetti);
            ELSE
                  SCRIVI("- Sta nello zaino o nella valigia, e non li
                  hai.");
            ENDIF
      ENDIF
      ENDIF
      IF(NOT INDOSSO_SPECIFICHE())THEN  // faccio alcuni controlli in
      indosso_specifiche
            IF(GET_LUOGO(GET_OGGETTO(og, oggetti)) = 0) THEN //controllo
            che l'oggetto si trovi nell'inventario
```

```
                        SCRIVI("Fatto.");
                  ENDIF
            ENDIF
      ENDIF
      ENDIF
      ENDIF
END
```

indossa_specifiche() (astro)

```
BEGIN
      problemi ← true;
      IF(og = 4 AND GET_LUOGO(GET_OGGETTO(22, oggetti)) = 0) THEN // controllo
      che l'oggetto sia la tuta e che il camice si trovi nell'inventario
            SCRIVI("Togli prima il camice.");
      ELSE IF(og = 22 AND GET_LUOGO(GET_OGGETTO(4, oggetti)) = 0) THEN //
      controllo che l'oggetto sia il camice e che la tuta si trovi
      nell'inventario
            SCRIVI("Togli prima la tuta.");
      ELSE
            problemi ← false;
      ENDIF
      ENDIF
      return (problemi);
END
```

indosso_specifiche() (astro)

```
BEGIN
      avvisato ← true;
      IF (og = 4 OR og = 11 OR og = 22) THEN //controllo che l'oggetto sia uno
      tra: tuta, casco e camice
            IF(GET_LUOGO(GET_OGGETTO(og, oggetti)) = 0) THEN //controllo che
      l'oggetto stia nell'inventario
                  SCRIVI("Ora l'hai addosso.");
            ENDIF
      ELSE
            avvisato = false;
      ENDIF
      return (avvisato);
END
```

Infine sono state apportate delle modifiche/aggiunte alle seguenti funzioni:

- azione_5 (save);
- azione_6 (load);
- azione_10;
- azione_15.

save () (gioco)

```
BEGIN
	salva ← true;
	SCRIVI("Salvataggio partita...");
	APRI_FILE();
	file ← salva;
	FOR (i = 1; i <= GET_N_OGGETTI(oggetti); i++)
		file ← GET_LUOGO(GET_OGGETTO(i, oggetti));
	ENDFOR
	file ← luogo_attuale;
	file ← tempo;
	file ← passo_soluzione;
	file ← n_oggettiZ;
	file ← peso_MaxZ;
	IF(NOT PILAVUOTA(PILA))THEN // controllo che lo zaino non sia vuoto
		el1  ←  LEGGIPILA(PILA);
		FUORIPILA(PILA);
		IF(NOT PILAVUOTA(PILA))THEN // controllo che lo zaino non sia vuoto
			el2  ←  LEGGIPILA(PILA);
			FUORIPILA(PILA);
			INPILA(el2, PILA);
		ENDIF
		INPILA(el1, PILA);
	ENDIF
	file ← el1;
	file ← el2;
	file ← n_oggettiV;
	file ← peso_MaxV;
	IF(NOT INSIEMEVUOTO(INSIEME))THEN//controllo che la valigia non sia vuota
		IF(APPARTIENE(50, INSIEME)) THEN // controllo che il casco sia
		presente nella valigia
			el3 ← 50;
		ENDIF
		file ← el3;
		IF(APPARTIENE(51, INSIEME)) THEN // controllo che la tuta sia
		presente nella valigia
			el4 ← 51;
		ENDIF
		file ← el4;
		IF(APPARTIENE(52, INSIEME)) THEN // controllo che il camice sia
		presente nella valigia
			el5 ← 52;
		ENDIF
		file ← el5;
		IF(APPARTIENE(55,INSIEME)) THEN // controllo che il manuale sia
		presente nella valigia
			el6 ← 55;
		ENDIF
		file ← el6;
	ENDIF
	SAVE_SPECIFICHE(FILE);
	CHIUDI_FILE(FILE);
END
```

load () (astro)

```
BEGIN
      APRI_FILE();
      file → salva;
      IF(salva) THEN // controllo che la variabile salva sia impostata a
      true(viene impostata quando si entra in save())
            SCRIVI("Caricamento partita...");
            FOR (i = 1; i <= GET_N_OGGETTI(oggetti); i++)
                  file → valore;
                  SET_LUOGO(i, valore, oggetti);
            ENDFOR
            file → luogo_attuale;
            file → tempo;
            file → passo_soluzione;
            WHILE(NOT PILAVUOTA(PILA))DO //controllo che lo zaino non sia vuoto
                  FUORIPILA (PILA);
            ENDWHILE
            file → n_oggettiZ;
            file → peso_MaxZ;
            file → el1;
            file → el2;
            IF(el2 !=0) THEN
                  INPILA(el2, PILA);
            ENDIF
            IF(el1 !=0) THEN
                  INPILA(el1, PILA);
            ENDIF
            file → n_oggettiV;
            file → peso_MaxV;
            file → el3;
            file → el4;
            file → el5;
            file → el6;
            IF((NOT(APPARTIENE(el3, INSIEME))) AND el3 != 0) THEN
                  INSERISCI(el3, INSIEME);
            ENDIF
            IF((NOT(APPARTIENE(el4, INSIEME))) AND el4 != 0) THEN
                  INSERISCI(el4, INSIEME);
            ENDIF
            IF((NOT(APPARTIENE(el5, INSIEME))) AND el5 != 0) THEN
                  INSERISCI(el5, INSIEME);
            ENDIF
            IF((NOT(APPARTIENE(el6, INSIEME))) AND el6 != 0) THEN
                  INSERISCI(el6, INSIEME);
            ENDIF
            LOAD_SPECIFICHE(FILE);
            CHIUDI_FILE(FILE);
      ELSE
            SCRIVI("Non e' presente nessun salvataggio...");
      ENDIF
END
```

azione_10 () (astro)

```
BEGIN
      IF (GET_LUOGO(GET_OGGETTO(og, oggetti)) != 0 AND GET_LUOGO(GET_OGGETTO(og,
      oggetti)) != 12) THEN // controllo che l'oggetto non si trovi né
      nell'inventario, né nello zaino, né nella valigia
            SCRIVI("- Non ce l'hai.");
      ELSE IF (NOT(luogo_attuale = 9) OR (luogo_attuale = 7 AND
      parete_stagna_aperta = 1)) THEN //controllo che il luogo attuale non sia
      la prua dell'astronave o che il luogo attuale sia il compartimento stagno
      con la combinazione di pulsanti(rosso-verde) settata ad 1, quindi pulsante
      rosso premuto
            LASCIA();
      ELSE
            SCRIVI("L'aria! L'aria! Aaaagh!!!");
            MORTO();
      ENDIF
      ENDIF
END
```

azione_15 () (astro)

```
BEGIN
      IF (GET_LUOGO(GET_OGGETTO(og, oggetti)) = luogo_attuale) THEN // controllo
      che l'oggetto si trovi nel luogo attuale
            SCRIVI("Prendilo in mano, prima.");
      ELSE IF(GET_LUOGO(GET_OGGETTO(og, oggetti)) = 12) THEN // controllo che
      l'oggetto si trovi nello zaino o nella valigia
            IF(APPARTIENE(55, INSIEME)) THEN // controllo che l'oggetto manuale
            si trovi nella valigia
                  SCRIVI("- MANUALE DI ISTRUZIONI DEL -");
                  SCRIVI("-   REATTORE POSITRONICO    -");
                  SCRIVI("-     Mod. YTREWQ 8421      ");
                  SCRIVI("- Per attivare il reattore,");
                  SCRIVI("tirare la leva e poi premere");
                  SCRIVI("in sequenza i pulsanti verde,");
                  SCRIVI("giallo e rosso.");
                  SCRIVI("- Per disattivare il reattore...");
                  SCRIVI("Dannazione! La pagina e' strappata.");
            ELSE IF(LEGGIPILA(PILA) = 55) THEN // controllo che l'oggetto
            manuale sia il primo visibile nello zaino
                  SCRIVI("- MANUALE DI ISTRUZIONI DEL -");
                  SCRIVI("-   REATTORE POSITRONICO    -");
                  SCRIVI("-     Mod. YTREWQ 8421      ");
                  SCRIVI("- Per attivare il reattore,");
                  SCRIVI("tirare la leva e poi premere");
                  SCRIVI("in sequenza i pulsanti verde,");
                  SCRIVI("giallo e rosso.");
                  SCRIVI("- Per disattivare il reattore...");
                  SCRIVI("Dannazione! La pagina e' strappata.");
            ELSE
                  SCRIVI("Devi prima lasciare: ");
                  GET_ZAINO(LEGGIPILA(PILA), oggetti);
            ENDIF
            ENDIF
      ELSE
            SCRIVI("- MANUALE DI ISTRUZIONI DEL -");
```

```
            SCRIVI("-   REATTORE POSITRONICO    -");
            SCRIVI("-     Mod. YTREWQ 8421      ");
            SCRIVI("- Per attivare il reattore,");
            SCRIVI("tirare la leva e poi premere");
            SCRIVI("in sequenza i pulsanti verde,");
            SCRIVI("giallo e rosso.");
            SCRIVI("- Per disattivare il reattore...");
            SCRIVI("Dannazione! La pagina e' strappata.");
      ENDIF
      ENDIF
END
```

3. REALIZZAZIONE

3.1 - PILA

Header della pila:

```
#ifndef ZAINO_H_
#define ZAINO_H_
#include <iostream>
using namespace std;
#include "nodoZ.h"

template <class T>
class Pila{
public:
      typedef Nodo<T>* posizione;
      typedef T tipoelem;
      Pila(); // costruttori
      ~Pila(); // distruttore
      void creaPila(); //operatori
      bool pilaVuota();
      tipoelem leggiPila();
      void fuoriPila();
      void inPila(tipoelem);
private:
      posizione pos;
};
```

Dei vari metodi:

- **Pila** e **~Pila** sono **costruttore** e **distruttore** e servono per gestire gli oggetti creati;
- **creaPila** serve per creare la pila;
- **pilaVuota** indica se la pila è vuota o meno;

- **leggiPila** restituisce il primo valore nella pila, cioè l'ultimo inserito;
- **fuoriPila** elimina dalla pila il primo valore che incontra, cioè l'ultimo inserito;
- **inPila** serve ad inserire un nuovo elemento all'interno della pila.

```
template <class T> Pila<T>::Pila(){
	creaPila();
}

template <class T> Pila<T>::~Pila(){}

template <class T> void Pila<T>::creaPila(){
	pos= new Nodo<T>;
	pos->scriviSucc(NULL);// successivo non esiste quando creo un nodo, quindi
	NULL
}

template <class T> bool Pila<T>::pilaVuota(){
	return(pos->leggiSucc()==NULL);
}

template <class T> T Pila<T>::leggiPila(){
	if (!pilaVuota())
		return(pos->leggiSucc()->leggiNodo()); // Valore che ritorna se la
		pila non è vuota
	else{
		//cout << "\nLa pila e' vuota...\n" << endl;
		return(0); // Valore che ritorna se la pila è vuota
	}
}

template <class T> void Pila<T>::fuoriPila(){
	if (!pilaVuota()){
		posizione temp = new Nodo<T>;
		temp->scriviSucc(pos->leggiSucc()->leggiSucc());
		delete(pos->leggiSucc());
		pos->scriviSucc(temp->leggiSucc());
		delete(temp);
	}
	/*else {
		cout<<"\nNessun elemento nella pila..\n"<<endl;
	}*/
}

template <class T> void Pila<T>::inPila(tipoelem el){
	posizione temp = new Nodo<T>;
	temp->scriviSucc(pos->leggiSucc());
	pos->scriviSucc(temp);
	temp->scriviNodo(el);
}

#endif /* ZAINO_H_ */
```

3.2 - INSIEME

Header della pila:

```
#ifndef VALIGIA_H_
#define VALIGIA_H_
#include <iostream>
using namespace std;
#include "listaV.h"

template <class T>
class Insieme{
public:
	typedef T tipoelem;
	Insieme();
	~Insieme();
	void creaInsieme();
	bool InsiemeVuoto();
	bool Appartiene(tipoelem);
	void Inserisci(tipoelem);
	void Cancella(tipoelem);
	//Insieme<T> Unione(Insieme<T>&);
	//Insieme<T> Intersezione(Insieme<T>&);
	//Insieme<T> Differenza(Insieme<T>&);
	Insieme<T> operator = (const Insieme<T>&);
private:
	ListaV<T> lista;
};
```

Dei vari metodi:

- **Insieme** e **~Insieme** sono **costruttore** e **distruttore** e servono per gestire gli oggetti creati;
- **creaInsieme** serve per creare l'insieme;
- **insiemeVuoto** indica se l'insieme è vuoto o meno;
- **Appartiene** restituisce true se l'elemento si trova nell'insieme, false altrimenti;
- **Inserisci** serve ad inserire un nuovo elemento all'interno dell'insieme.
- **Cancella** elimina dall'insieme l'elemento desiderato.

```
template <class T> Insieme<T>::Insieme(){creaInsieme();};

template <class T> Insieme<T>::~Insieme(){};

template <class T> void Insieme<T>::creaInsieme(){
	lista.creaLista();
}

template <class T> bool Insieme<T>::InsiemeVuoto(){
	return(lista.listaVuota());
```

```
}

template <class T> bool Insieme<T>::Appartiene(tipoelem a){
	CellaV<T>* indice=lista.primoLista();
	bool trovato=false;
	if(!lista.listaVuota()){
		do
		{
			indice = lista.succLista(indice);
			if(a==lista.leggiLista(indice)){
				trovato=true;
			}
		}while (!lista.fineLista(indice) && !trovato);
		//return(trovato);
	}
	return(trovato);
}

template <class T> void Insieme<T>::Inserisci(tipoelem a){
	CellaV<T>* indice=lista.primoLista();  //è l'indice della lista ordinata
    if(lista.listaVuota())
    {
      lista.insLista(indice, a);
    }
        else if(!Appartiene(a))
        {
            while(!lista.fineLista(indice) &&
a>lista.leggiLista(lista.succLista(indice)))
            {
                indice=lista.succLista(indice);
            }
            lista.insLista(indice, a);
        }
}

template <class T> void Insieme<T>::Cancella(tipoelem a){
	CellaV<T>* indice=lista.primoLista();
	if(Appartiene(a)){
		indice = lista.succLista(indice);
		while (!lista.fineLista(indice) && a!=lista.leggiLista(indice)){
			indice=lista.succLista(indice);
		}
		lista.cancLista(indice);
	}
}

/*template <class T> Insieme<T> Insieme<T>::Unione(Insieme<T>& i2){
	CellaV<T>* indice=lista.primoLista();
	Insieme<T> temp;
	temp=i2;
	do
	{
		indice = lista.succLista(indice);
		temp.Inserisci(lista.leggiLista(indice));
	}while (!lista.fineLista(indice));
	return temp;
}

template <class T> Insieme<T> Insieme<T>::Intersezione(Insieme<T>& i2){
```

```
	CellaV<T>* indice=lista.succLista(lista.primoLista());
	Insieme<T> temp;
	do
	{
		if(i2.Appartiene(lista.leggiLista(indice)))
			temp.Inserisci(lista.leggiLista(indice));
		indice=lista.succLista(indice);
	}while (!lista.fineLista(indice));
	if(i2.Appartiene(lista.leggiLista(indice)))
		temp.Inserisci(lista.leggiLista(indice));
	return temp;
}

template <class T> Insieme<T> Insieme<T>::Differenza(Insieme<T>& i2){
	CellaV<T>* indice=lista.primoLista();
	Insieme<T> temp;
	//temp=i2; //B-A
	temp=*this; //A-B
	do
	{
		indice = lista.succLista(indice);
		if(i2.Appartiene(lista.leggiLista(indice)))
			temp.Cancella(lista.leggiLista(indice));
	}while (!lista.fineLista(indice));
	return temp;
}*/
template <class T> Insieme<T> Insieme<T>::operator=(const Insieme<T> &i2){
	CellaV<T>* indice=i2.lista.primoLista();

	if(!i2.lista.listaVuota())
	{
		do{
			indice=i2.lista.succLista(indice);
			Inserisci(i2.lista.leggiLista(indice));
		}while(!i2.lista.fineLista(indice));
	}
	return *this;
}
#endif /* VALIGIA_H_ */
```

3.3 - VOCABOLARIO

Nel vocabolario presente nella classe “Astro” ho inserito le seguenti parole:

- vocabolario.inserisci (“aiuto”, 58);
- vocabolario.inserisci (“help”, 58);
- vocabolario.inserisci (“valigia”, 56);
- vocabolario.inserisci (“zaino”, 57).

3.4 - OGGETTI

Ho inserito inoltre i seguenti oggetti:

- oggetti.inserisci(Oggetto(“una valigia”, 56, 5, 0));
- oggetti.inserisci(Oggetto(“uno zaino”, 57, 6, 0));

3.5 - AZIONI

Ho inserito le seguenti azioni:

- azioni.inserisci(856, -2) // codice azione prendi valigia
- azioni.inserisci(857, -2) // codice azione prendi zaino
- azioni.inserisci(956, 3) // codice azione lascia valigia
- azioni.inserisci(957, 3) // codice azione lascia zaino
- azioni.inserisci(1056, -39) // codice azione guarda valigia
- azioni.inserisci(1057, -40) // codice azione guarda zaino
- azioni.inserisci(5600, 44) // codice azione inventario valigia
- azioni.inserisci(5700, 42) // codice azione inventario zaino
- azioni.inserisci(5800, 43) // codice azione aiuto/help

E modificato queste altre:

- azioni.inserisci(2050, -2) //codice azione indossa/metti casco
- azioni.inserisci(2051, -2) // codice azione indossa/metti tuta
- azioni.inserisci(2052, -2) // codice azione indossa/metti camice

in

- azioni.inserisci(2050, -41) // codice azione indossa/metti casco
- azioni.inserisci(2051, -41) // codice azione indossa/metti tuta
- azioni.inserisci(2052, -41) // codice azione indossa/metti camice

Di seguito la loro realizzazione:

```
void Gioco::prendi() {
      if (oggetti.get_oggetto(og).get_luogo() == 0)
            interfaccia.scrivi("- Gia' fatto.");
      else if(oggetti.get_oggetto(og).get_luogo() == 12)
            interfaccia.scrivi("- Sta gia' nello zaino o nella valigia.");
```

```
	else if (oggetti.get_oggetto(og).get_luogo() < 0)
		interfaccia.scrivi("- Non e' possibile.");
	else if (!prendi_specifiche()) {
		if(oggetti.get_oggetto(25).get_luogo() == 0){
			if (og == 20 || og == 23)
				oggetti.set_luogo(og,0);
			else
				oggetti.set_luogo(og,12);
		}
		else if(oggetti.get_oggetto(26).get_luogo() == 0){
			if (og == 20 || og == 23)
				oggetti.set_luogo(og,0);
			else
				oggetti.set_luogo(og,12);
		}
		else
			oggetti.set_luogo(og,0);
		if (!preso_specifiche())
		   interfaccia.scrivi("Fatto.");
	}
}

bool Astro::prendi_specifiche() {
	bool problemi = true;

	if ((og == 4 || og == 11 || og == 22 ) &&
	(oggetti.get_oggetto(25).get_luogo() != 0 &&
	oggetti.get_oggetto(26).get_luogo() != 0)){
		interfaccia.scrivi("Trova prima uno zaino o una valigia.");
		interfaccia.scrivi("(Suggerimento: puoi comunque indossarlo.)");
	}
	else if(og == 25 && oggetti.get_oggetto(26).get_luogo() == 0){
		interfaccia.scrivi("Lascia prima lo zaino.");
	}
	else if(og == 26 && oggetti.get_oggetto(25).get_luogo() == 0){
		interfaccia.scrivi("Lascia prima la valigia.");
	}
	else
		problemi = false;
	return problemi;
}

bool Astro::preso_specifiche() {
	bool avvisato = true;

	if ((og == 4 || og == 11 || og == 22 || og == 24) &&
	(oggetti.get_oggetto(25).get_luogo() == 0)){
		if(oggetti.get_oggetto(og).get_peso() > peso_MaxV && og != 24){
			oggetti.set_luogo(og,luogo_attuale);
			interfaccia.scrivi("La valigia e' troppo piena...");
			interfaccia.scrivi("(Suggerimento: togli qualcosa dalla
			valigia.)");
		}
		else if(oggetti.get_oggetto(og).get_peso() > peso_MaxV && og == 24){
			oggetti.set_luogo(og,0);
			interfaccia.scrivi("La valigia e' troppo piena...");
			interfaccia.scrivi("Lo tengo in mano, e' troppo importante.");
			interfaccia.scrivi("(Suggerimento: controlla
			nell'inventario)");
```

```
			}
			else{
				interfaccia.scrivi("Ora e' in valigia.");
				ins.Inserisci(oggetti.get_oggetto(og).get_codice());
				n_oggettiV++;
				peso_MaxV -= oggetti.get_oggetto(og).get_peso();
			}
		}
		else if ((og == 4 || og == 11 || og == 22 || og == 24) &&
		(oggetti.get_oggetto(26).get_luogo() == 0)){
			if(oggetti.get_oggetto(og).get_peso() > peso_MaxZ && og != 24){
				oggetti.set_luogo(og,luogo_attuale);
				interfaccia.scrivi("Lo zaino e' troppo pieno...");
				interfaccia.scrivi("(Suggerimento: togli qualcosa dallo
				zaino.)");
			}
			else if(oggetti.get_oggetto(og).get_peso() > peso_MaxZ && og == 24){
				oggetti.set_luogo(og,0);
				interfaccia.scrivi("Lo zaino e' troppo pieno...");
				interfaccia.scrivi("Lo tengo in mano, e' troppo importante.");
				interfaccia.scrivi("(Suggerimento: controlla
				nell'inventario)");
			}
			else{
				interfaccia.scrivi("Ora e' nello zaino.");
				p.inPila(oggetti.get_oggetto(og).get_codice());
				n_oggettiZ++;
				peso_MaxZ -= oggetti.get_oggetto(og).get_peso();
			}
		}
		else
			avvisato = false;
		return avvisato;
}

void Gioco::lascia() {
		if (og == 0 && oggetti.get_oggetto(og).get_luogo() != 0 &&
		oggetti.get_oggetto(og).get_luogo() != 12)
			interfaccia.scrivi("- Non ce l'hai e qui non c'e'.");
		else if(og != 0 && oggetti.get_oggetto(og).get_luogo() != 0 &&
		oggetti.get_oggetto(og).get_luogo() != 12)
			interfaccia.scrivi("- Non ce l'hai.");
		else if (og == 0 && oggetti.get_oggetto(og).get_luogo() != 0)
			interfaccia.scrivi("- Non ce l'hai.");
		else if (!lascia_specifiche()) {
			if(oggetti.get_oggetto(26).get_luogo() == 0){
				if (oggetti.get_oggetto(4).get_luogo() == 12 ||
				oggetti.get_oggetto(11).get_luogo() == 12 ||
				oggetti.get_oggetto(22).get_luogo() == 12 ||
				oggetti.get_oggetto(24).get_luogo() == 12){
					if(og == 4 || og == 11 || og == 22 || og == 24){
						if(oggetti.get_oggetto(og).get_luogo()== 0){
							oggetti.set_luogo(og,luogo_attuale);
							interfaccia.scrivi("Fatto.");
						}
						else{
							int c,d;
							c=p.leggiPila();
							d=og;
```

```
                        if(og == oggetti.get_zaino2(c)){
                              oggetti.set_luogo(og,luogo_attuale);
                              p.fuoriPila();
                              n_oggettiZ--;
                              peso_MaxZ +=
                              oggetti.get_oggetto(og).get_peso();
                              interfaccia.scrivi("Fatto.");
                        }
                        else
                        if(ins.Appartiene(oggetti.get_valigia(d))){
                              interfaccia.scrivi("- Non ce l'hai
                              addosso e non sta nello zaino..");
                              interfaccia.scrivi(" (Suggerimento:
                              controlla nella valigia)");
                        }
                        else if (!p.pilaVuota()){
                              interfaccia.scrivi("Devi prima
                              lasciare: ");
                              oggetti.get_zaino(c);
                        }
                  }
            }
            else{
                  oggetti.set_luogo(og,luogo_attuale);
                  interfaccia.scrivi("Fatto.");
            }
      }
      else{
            if(oggetti.get_oggetto(og).get_luogo() != 0){
                  interfaccia.scrivi("- Non ce l'hai.");
            }
            else{
                  oggetti.set_luogo(og,luogo_attuale);
                  interfaccia.scrivi("Fatto.");
            }
      }
}
else if(oggetti.get_oggetto(25).get_luogo() == 0){
      if (oggetti.get_oggetto(4).get_luogo() == 12 ||
      oggetti.get_oggetto(11).get_luogo() == 12 ||
      oggetti.get_oggetto(22).get_luogo() == 12 ||
      oggetti.get_oggetto(24).get_luogo() == 12){
            if(og == 4 || og == 11 || og == 22 || og == 24){
                  int c;
                  c=og;
                  if(oggetti.get_oggetto(og).get_luogo()== 0){
                        oggetti.set_luogo(og,luogo_attuale);
                        interfaccia.scrivi("Fatto.");
                  }
                  else if(!ins.Appartiene(oggetti.get_valigia(c))){
                        if((oggetti.get_valigia(c))==(p.leggiPila())
                        ){
                              interfaccia.scrivi("- Non ce l'hai in
                              valigia.");
                              interfaccia.scrivi(" (Suggerimento:
                              controlla nello zaino)");
                        }
```

```
                                    else
                                    if((oggetti.get_valigia(c))!=(p.leggiPila())
                                    )
                                          interfaccia.scrivi("- Non ce l'hai in
                                          valigia.");
                              }
                              else if(ins.Appartiene(oggetti.get_valigia(c))){
                                    oggetti.set_luogo(og,luogo_attuale);
                                    ins.Cancella(oggetti.get_valigia(c));
                                    n_oggettiV--;
                                    peso_MaxV +=
                                    oggetti.get_oggetto(og).get_peso();
                                    interfaccia.scrivi("Fatto.");
                              }
                              else
                                    interfaccia.scrivi("- Non ce l'hai in
                                    valigia.");
                        }
                        else{
                              oggetti.set_luogo(og,luogo_attuale);
                              interfaccia.scrivi("Fatto.");
                        }
                  }
                  else{
                        if(oggetti.get_oggetto(og).get_luogo() != 0){
                              interfaccia.scrivi("- Non ce l'hai.");
                        }
                        else{
                              oggetti.set_luogo(og,luogo_attuale);
                              interfaccia.scrivi("Fatto.");
                        }
                  }
            }
            else{
                  if(oggetti.get_oggetto(og).get_luogo() == 0){
                        oggetti.set_luogo(og,luogo_attuale);
                        interfaccia.scrivi("Fatto.");
                  }
                  else
                        interfaccia.scrivi("- Non ce l'hai.");
            }
      }
      else if ((luogo_attuale == 7 || luogo_attuale >= 9) &&
      (oggetti.get_oggetto(4).get_luogo() != 0 ||
      oggetti.get_oggetto(11).get_luogo() != 0)) {
            interfaccia.scrivi("\nAaaagh!!!");
            morto();
      }
}

bool Astro::lascia_specifiche() {
      bool lasciato = true;

      if (luogo_attuale >= 9 || (luogo_attuale == 7 && parete_stagna_aperta==1))
      {
            oggetti.set_luogo(og, -99);
            interfaccia.scrivi("Si e' perso nello spazio.");
      }
      else
```

```
            lasciato = false;
      return lasciato;
}

void Oggetti::get_zaino(int c) {
      bool trovato = false;
      int i = 1;
      while (i <= fine_Oggetti && !trovato) {
            if (oggetti[i].get_codice() == c){
                  cout << "- " <<oggetti[i].get_nome() << endl;
                  trovato = true;
            }i++;
      }
}

int Oggetti::get_zaino2(int c) {
      bool trovato = false;
      int i = 1;
      int i2 = 0;
      while (i <= fine_Oggetti && !trovato) {
            if (oggetti[i].get_codice() == c){
                  i2=i;
                  trovato = true;
            }
            i++;
      }
      return(i2);
}

int Oggetti::get_valigia(int c) {
      bool trovato = false;
      int i = 1;
      int i2 = 0;
      while (i <= fine_Oggetti && !trovato) {
            if (i == c){
                  i2= oggetti[i].get_codice();
                  trovato = true;
            }
            i++;
      }
      return(i2);
}

void Astro::azione_39() {
      interfaccia.scrivi("E' la valigia del secondo pilota.");
      interfaccia.scrivi("Sembra molto capiente.");
}

void Astro::azione_40() {
      interfaccia.scrivi("E' il tuo zaino, puo' essere utile per trasportare
      oggetti.");
}

void Astro::azione_44() {
      if(oggetti.get_oggetto(25).get_luogo() == 0){
            if (!ins.InsiemeVuoto()) {
                  interfaccia.scrivi("Inventario Valigia");
                  interfaccia.scrivi("\nVedo: ");
                  if(ins.Appartiene(50))
```

```
                        oggetti.get_zaino(50);
                  if(ins.Appartiene(51))
                        oggetti.get_zaino(51);
                  if(ins.Appartiene(52))
                        oggetti.get_zaino(52);
                  if(ins.Appartiene(55))
                        oggetti.get_zaino(55);
                  cout<<"\nTotale Oggetti nella Valigia: "<< n_oggettiV << endl;
                  cout<<"Spazio disponibile:"<< peso_MaxV << " su 8 kg."<< endl;
            }else
                  interfaccia.scrivi("E' vuota.");
      }
      else
            interfaccia.scrivi("Non ce l'hai.");
}

void Astro::azione_42() {
      if(oggetti.get_oggetto(26).get_luogo() == 0){
            if (!p.pilaVuota()) {
                  interfaccia.scrivi("Inventario Zaino");
                  interfaccia.scrivi("\nVedo in cima: ");
                  int c;
                  c=p.leggiPila();
                  oggetti.get_zaino(c);
                  cout<< "\nTotale Oggetti nello Zaino: " << n_oggettiZ << endl;
                  cout<<"Spazio disponibile:"<< peso_MaxZ << " su 5 kg."<< endl;
            }
            else
                  interfaccia.scrivi("E' vuoto.");
      }
      else
            interfaccia.scrivi("Non lo hai.");
}

void Astro::azione_43() {
      interfaccia.scrivi("\nCOMANDI DI GIOCO:");
      interfaccia.scrivi("\nDirezioni: ");
      interfaccia.scrivi("- n/nord: per muoverti in avanti;");
      interfaccia.scrivi("- s/sud: per muoverti indietro;");
      interfaccia.scrivi("- e/est: per muoverti a destra;");
      interfaccia.scrivi("- w/o/ovest: per muoverti a sinistra;");
      interfaccia.scrivi("- a/alto/sali: per salire ad un piano superiore;");
      interfaccia.scrivi("- b/basso/scendi: per scendere ad un piano
      inferiore;");
      interfaccia.scrivi("\nAzioni: ");
      interfaccia.scrivi("- prendi: per trasportare un oggetto in mano o con
      se(zaino/valigia);");
      interfaccia.scrivi("- indossa/metti: per indossare un oggetto(es.
      casco);");
      interfaccia.scrivi("- guarda: per guardare ed ottenere informazioni su un
      oggetto (es.tuta);");
      interfaccia.scrivi("- lascia/togli/leva: per lasciare o togliersi gli
      oggetti trasportati;");
      interfaccia.scrivi("- apri: per aprire un oggetto fisso(es.
      armadietto);");
      interfaccia.scrivi("- leggi: per leggere una scritta(es.cartello);");
      interfaccia.scrivi("- spingi/tira: per spingere o tirare un oggetto
      fisso(es.leva);");
```

```
	interfaccia.scrivi("- premi/schiaccia: per premere un oggetto
	fisso(es.pulsante);");
	interfaccia.scrivi("- inventario/cosa: per accedere all'inventario degli
	oggetti trasportati;");
	interfaccia.scrivi("- zaino: per accedere agli oggetti trasportati nello
	zaino;");
	interfaccia.scrivi("- valigia: per accedere agli oggetti trasportati nella
	valigia;");
	interfaccia.scrivi("- save/load: per salvare o caricare la partita;");
}

void Astro::azione_41() {
	if (oggetti.get_oggetto(og).get_luogo() == 0)
		interfaccia.scrivi("- Gia' fatto.");
	else if (oggetti.get_oggetto(og).get_luogo() < 0)
		interfaccia.scrivi("- Non e' possibile.");
	else if (!indossa_specifiche()) {
		if(oggetti.get_oggetto(26).get_luogo() == 0){
			if (oggetti.get_oggetto(4).get_luogo() == 12 ||
			oggetti.get_oggetto(11).get_luogo() == 12 ||
			oggetti.get_oggetto(22).get_luogo() == 12){
				if(og == 4 || og == 11 || og == 22){
					int c;
					c=p.leggiPila();
					if(og == oggetti.get_zaino2(c)){
						oggetti.set_luogo(og,0);
						p.fuoriPila();
						n_oggettiZ--;
						peso_MaxZ +=
						oggetti.get_oggetto(og).get_peso();
					}
					else if(oggetti.get_oggetto(og).get_luogo() ==
					luogo_attuale){
						oggetti.set_luogo(og,0);
					}
					else{
						interfaccia.scrivi("Devi prima lasciare: ");
						oggetti.get_zaino(c);
					}
				}
				else{
					oggetti.set_luogo(og,0);
					interfaccia.scrivi("Fatto.");
				}
			}
			else{
				oggetti.set_luogo(og,0);
			}
		}
		else if(oggetti.get_oggetto(25).get_luogo() == 0){
			if (oggetti.get_oggetto(4).get_luogo() == 12 ||
			oggetti.get_oggetto(11).get_luogo() == 12 ||
			oggetti.get_oggetto(22).get_luogo() == 12){
				if(og == 4 || og == 11 || og == 22){
					int c;
					c=og;
					if(ins.Appartiene(oggetti.get_valigia(c))){
						oggetti.set_luogo(og,0);
						ins.Cancella(oggetti.get_valigia(c));
```

```
                                    n_oggettiV--;
                                    peso_MaxV +=
                                    oggetti.get_oggetto(og).get_peso();
                              }
                              else if(!ins.Appartiene(oggetti.get_valigia(c)))
                                    interfaccia.scrivi("- Non ce l'hai in
                                    valigia.");
                              else if(oggetti.get_oggetto(og).get_luogo() ==
                              luogo_attuale){
                                    oggetti.set_luogo(og,0);
                              }
                        }
                        else{
                              oggetti.set_luogo(og,0);
                              interfaccia.scrivi("Fatto.");
                        }
                  }
                  else{
                        oggetti.set_luogo(og,0);                        }
            }
            else{
                  if(og != 0 && oggetti.get_oggetto(og).get_luogo() != 12){
                        oggetti.set_luogo(og,0);
                  }
                  else
                        interfaccia.scrivi("- Sta nello zaino o nella valigia, e
                        non li hai.");
            }
            if (!indosso_specifiche()){
                  if(oggetti.get_oggetto(og).get_luogo() == 0)
                        interfaccia.scrivi("Fatto.");
            }
      }
}

bool Astro::indossa_specifiche() {
      bool problemi = true;

      if (og == 4 && oggetti.get_oggetto(22).get_luogo() == 0)
      interfaccia.scrivi("Togli prima il camice.");
      else if (og == 22 && oggetti.get_oggetto(4).get_luogo() == 0)
            interfaccia.scrivi("Togli prima la tuta.");
      else
            problemi = false;
      return problemi;
}

bool Astro::indosso_specifiche() {
      bool avvisato = true;

      if (og == 4 || og == 11 || og == 22)
            if(oggetti.get_oggetto(og).get_luogo() == 0)
                  interfaccia.scrivi("Ora l'hai addosso.");
      else
            avvisato = false;
      return avvisato;
}
```

Infine sono state apportate delle modifiche/aggiunte alle seguenti funzioni:

- azione_5 (save);
- azione_6 (load);
- azione_10;
- azione_15.

Di seguito la loro realizzazione:

```
void Gioco::save() {
	int i;
	salva=true;
	interfaccia.scrivi("Salvataggio partita...");
	ofstream file(Salvataggio, ios::out);
	for (i = 1; i <= oggetti.get_n_oggetti(); i++)
		file << oggetti.get_oggetto(i).get_luogo() << '\n';
	}
	file << salva << '\n';
	file << luogo_attuale << '\n';
	file << tempo << '\n';
	file << passo_soluzione << '\n';
	file << n_oggettiZ << '\n';
	file << peso_MaxZ << '\n';
	if(!(p.pilaVuota())){
		el1 = p.leggiPila();
		p.fuoriPila();
		if(!(p.pilaVuota())){
			el2 = p.leggiPila();
			p.fuoriPila();
			p.inPila(el2);
		}
		p.inPila(el1);
	}
	file << el1 << '\n';
	file << el2 << '\n';
	file << n_oggettiV << '\n';
	file << peso_MaxV << '\n';
	if(!(ins.InsiemeVuoto())){
		if(ins.Appartiene(50)){
			el3 = 50;
		}
		file << el3 << '\n';
		if(ins.Appartiene(51)){
			el4 = 51;
		}
		file << el4 << '\n';
		if(ins.Appartiene(52)){
			el5 = 52;
		}
		file << el5 << '\n';
		if(ins.Appartiene(55)){
			el6 = 55;
		}
		file << el6 << '\n';
	}
```

```
	save_specifiche(file);
	file.close();
}

void Gioco::load() {
	int i;
	int valore;
	interfaccia.scrivi("Caricamento partita...");
	ifstream file(Salvataggio, ios::in);
	for (i = 1; i <= oggetti.get_n_oggetti(); i++) {
	  file >> valore;
		oggetti.set_luogo(i,valore);
	}
	file >> salva;
	file >> luogo_attuale;
	file >> tempo;
	file >> passo_soluzione;
	if(salva){
		while(!(p.pilaVuota()))
			p.fuoriPila();
		file >> n_oggettiZ;
		file >> peso_MaxZ;
		file >> el1;
		file >> el2;
		if(el2 !=0)
			p.inPila(el2);
		if(el1 !=0)
			p.inPila(el1);

		file >> n_oggettiV;
		file >> peso_MaxV;
		file >> el3;
		file >> el4;
		file >> el5;
		file >> el6;
		if((!(ins.Appartiene(el3))) && el3 != 0)
			ins.Inserisci(el3);
		if((!(ins.Appartiene(el4))) && el4 != 0)
			ins.Inserisci(el4);
		if((!(ins.Appartiene(el5))) && el5 != 0)
			ins.Inserisci(el5);
		if((!(ins.Appartiene(el6))) && el6 != 0)
			ins.Inserisci(el6);
	}
	load_specifiche(file);
	file.close();
}

void Astro::azione_10() {
	if (oggetti.get_oggetto(og).get_luogo() != 0 &&
	oggetti.get_oggetto(og).get_luogo() != 12)
		interfaccia.scrivi("- Non ce l'hai.");
	else if (!(luogo_attuale == 9) || (luogo_attuale == 7 &&
	parete_stagna_aperta == 1))
		lascia();
	else {
		interfaccia.scrivi("L'aria! L'aria! Aaaagh!!!");
		morto();
	}
```

```
}

void Astro::azione_15() {
	if (oggetti.get_oggetto(og).get_luogo() == luogo_attuale)
		interfaccia.scrivi("Prendilo in mano, prima.");
	else if(oggetti.get_oggetto(og).get_luogo() == 12){
		if(ins.Appartiene(55)){
			interfaccia.scrivi("- MANUALE DI ISTRUZIONI DEL -");
			interfaccia.scrivi("-   REATTORE POSITRONICO    -");
			interfaccia.scrivi("-     Mod. YTREWQ 8421      -\n");
			interfaccia.scrivi("- Per attivare il reattore,");
			interfaccia.scrivi("tirare la leva e poi premere");
			interfaccia.scrivi("in sequenza i pulsanti verde,");
			interfaccia.scrivi("giallo e rosso.");
			interfaccia.scrivi("- Per disattivare il reattore...\n");
			interfaccia.scrivi("Dannazione! La pagina e' strappata.");
		}
		else if(p.leggiPila() == 55){
			interfaccia.scrivi("- MANUALE DI ISTRUZIONI DEL -");
			interfaccia.scrivi("-   REATTORE POSITRONICO    -");
			interfaccia.scrivi("-     Mod. YTREWQ 8421      -\n");
			interfaccia.scrivi("- Per attivare il reattore,");
			interfaccia.scrivi("tirare la leva e poi premere");
			interfaccia.scrivi("in sequenza i pulsanti verde,");
			interfaccia.scrivi("giallo e rosso.");
			interfaccia.scrivi("- Per disattivare il reattore...\n");
			interfaccia.scrivi("Dannazione! La pagina e' strappata.");
		}
		else{
			interfaccia.scrivi("Devi prima lasciare: ");
			oggetti.get_zaino(p.leggiPila());
		}
	}
	else{
		interfaccia.scrivi("- MANUALE DI ISTRUZIONI DEL -");
		interfaccia.scrivi("-   REATTORE POSITRONICO    -");
		interfaccia.scrivi("-     Mod. YTREWQ 8421      -\n");
		interfaccia.scrivi("- Per attivare il reattore,");
		interfaccia.scrivi("tirare la leva e poi premere");
		interfaccia.scrivi("in sequenza i pulsanti verde,");
		interfaccia.scrivi("giallo e rosso.");
		interfaccia.scrivi("- Per disattivare il reattore...\n");
		interfaccia.scrivi("Dannazione! La pagina e' strappata.");
	}
}
```

4. ESEMPI

Di seguito alcuni esempi grafici delle modifiche apportate:

```
Sei nella tua cabina.
Vedo:
- un letto;
- un armadietto;
- un casco;
- uno zaino.

Cosa devo fare?
guarda zaino

E' il tuo zaino, puo' essere utile per trasportare oggetti.

Sei nella tua cabina.
Vedo:
- un letto;
- un armadietto;
- un casco;
- uno zaino.

Cosa devo fare?
prendi casco

Trova prima uno zaino o una valigia.
(Suggerimento: puoi comunque indossarlo.)
```

```
Sei nella tua cabina.
Vedo:
- un letto;
- un armadietto;
- un casco;
- uno zaino.

Cosa devo fare?
prendi zaino

Fatto.

Sei nella tua cabina.
Vedo:
- un letto;
- un armadietto;
- un casco.

Cosa devo fare?
zaino

E' vuoto.
```

```
Sei nella tua cabina.
Vedo:
- un letto;
- un armadietto;
- un casco.

Cosa devo fare?
prendi casco

Ora e' nello zaino.

Sei nella tua cabina.
Vedo:
- un letto;
- un armadietto.

Cosa devo fare?
zaino

Inventario Zaino

Vedo in cima:
- un casco

Totale Oggetti nello Zaino: 1
Spazio disponibile: 2 su 5 kg.

Sei nella tua cabina.
Vedo:
- un letto;
- un armadietto.

Cosa devo fare?
lascia casco

Fatto.

Sei nella tua cabina.
Vedo:
- un letto;
- un armadietto;
- un casco.

Cosa devo fare?
zaino

E' vuoto.
```

```
Sei nella cabina del secondo pilota.
Vedo:
- un letto;
- un armadietto;
- un casco;
- un camice;
- un manuale;
- una valigia.

Cosa devo fare?
prendi casco

Ora e' nello zaino.

Sei nella cabina del secondo pilota.
Vedo:
- un letto;
- un armadietto;
- un camice;
- un manuale;
- una valigia.

Cosa devo fare?
zaino

Inventario Zaino

Vedo in cima:
- un casco

Totale Oggetti nello Zaino: 2
Spazio disponibile: 0 su 5 kg.

Sei nella cabina del secondo pilota.
Vedo:
- un letto;
- un armadietto;
- un camice;
- un manuale;
- una valigia.

Cosa devo fare?
prendi camice

Lo zaino e' troppo pieno...
(Suggerimento: togli qualcosa dallo zaino.)
```

```
Vedo:
- un letto;
- un armadietto;
- un camice;
- un manuale;
- una valigia.

Cosa devo fare?
lascia tuta

Devi prima lasciare:
- un casco
```

```
Sei nella cabina del secondo pilota.
Vedo:
- un letto;
- un armadietto;
- un camice;
- un manuale;
- una valigia.

Cosa devo fare?
lascia casco

Fatto.

Sei nella cabina del secondo pilota.
Vedo:
- un letto;
- un armadietto;
- un casco;
- un camice;
- un manuale;
- una valigia.

Cosa devo fare?
zaino

Inventario Zaino

Vedo in cima:
- una tuta

Totale Oggetti nello Zaino: 1
Spazio disponibile: 3 su 5 kg.
```

```
Cosa devo fare?
zaino

Inventario Zaino

Vedo in cima:
- un casco

Totale Oggetti nello Zaino: 2
Spazio disponibile: 0 su 5 kg.

Sei nella cabina del secondo pilota.
Vedo:
- un letto;
- un armadietto;
- un camice;
- un manuale;
- una valigia.

Cosa devo fare?
prendi manuale

Lo zaino e' troppo pieno...
Lo tengo in mano, e' troppo importante.
(Suggerimento: controlla nell'inventario)

Sei nella cabina del secondo pilota.
Vedo:
- un letto;
- un armadietto;
- un camice;
- una valigia.

Cosa devo fare?
cosa

Possiedi:
- il secondo pilota;
- una chiave;
- un manuale;
- uno zaino.
```

```
Cosa devo fare?
guarda valigia

E' la valigia del secondo pilota.
Sembra molto capiente.

Sei nella cabina del secondo pilota.
Vedo:
- una tuta;
- un letto;
- un armadietto;
- un casco;
- un camice;
- un manuale;
- una valigia;
- uno zaino.

Cosa devo fare?
prendi valigia

Fatto.

Sei nella cabina del secondo pilota.
Vedo:
- una tuta;
- un letto;
- un armadietto;
- un casco;
- un camice;
- un manuale;
- uno zaino.

Cosa devo fare?
prendi casco

Ora e' in valigia.

Sei nella cabina del secondo pilota.
Vedo:
- una tuta;
- un letto;
- un armadietto;
- un camice;
- un manuale;
- uno zaino.

Cosa devo fare?
valigia

Inventario Valigia

Vedo:
- un casco

Totale Oggetti nella Valigia: 1
Spazio disponibile: 5 su 8 kg.
```