INTRODUCTION

- With just the elementary mathematics of sets encountered so far we can describe a *simple* computerized system

- Of necessity, the system is trivial, but will enable us to give a taste of how mathematics could be used to specify a system:

  An aircraft has a fixed capacity and it is required to record the number of people aboard the aircraft at any time. The aircraft seats are not numbered and passengers enter the aircraft and choose seats on a first-come-first-served basis.

- Passengers belong to the set of all possible persons. Let this set be called *PERSON*.

- Hence, for this system, the *basic type* (or *given set*) is:

[PERSON]	the set of all possible uniquely identified persons

SYSTEM STATE

- If *capacity* denotes the seating capacity of the aircraft we have:

  capacity : $\mathbb{N}$ - the seating capacity of the aircraft

- At any time the *state* of the system is given by the number of passengers on the aircraft. We can describe this state by a set of persons, *onboard* (which will be one of the many possible subsets of *PERSON*)

- Hence we have: *onboard* : $\mathbb{P}$ PERSON

- In addition we have the obvious constraint:

  $$\#onboard \leq capacity$$

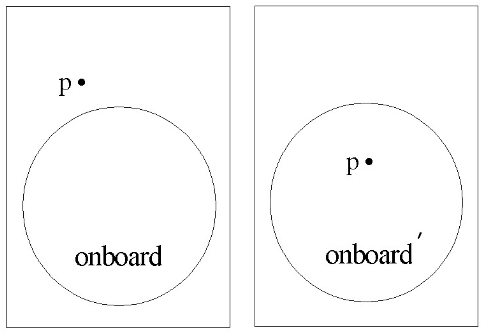
  - This constraint is an *invariant* for the system since, no matter what changes occur, it must still be true

OPERATIONS AFFECTING STATE

- Before passengers board the aircraft we will have the *initial state*:

  onboard = { } (which satisfies the invariant since #{ } = 0)

- When a passenger, *p*, boards the aircraft the set *onboard* will change. The value of *onboard* after such a change is written as *onboard'* (read as "*onboard prime*")



Before ***After***

- Note: $onboard' = onboard \cup \{p\}$

  [also, by implication: $\# onboard' = \# onboard + 1$]

OPERATIONS AFFECTING STATE

- Clearly, for the invariant to be satisfied it is essential that before a passenger can board the aircraft, the following *precondition* must be met:

$$\# \mathit{onboard} < \mathit{capacity}$$

- If the person boarding is *p* then a further *precondition* must be:

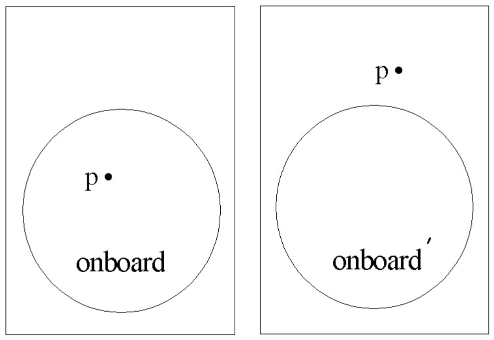
$$p \notin \mathit{onboard}$$

- Hence the boarding operation may be defined by

$$\begin{array}{l} p : \mathrm{PERSON} \\ p \notin \mathit{onboard} \\ \# \mathit{onboard} < \mathit{capacity} \\ \mathit{onboard}' = \mathit{onboard} \cup \{p\} \end{array}$$

OPERATIONS AFFECTING STATE

- Passengers will also be able to disembark from the aircraft:



Before ***After***

- After disembarkation the following must be true:

$$onboard' = onboard \setminus \{p\}$$

with *precondition*: $p \in onboard$

- Summarizing: p **:** PERSON
  $p \in onboard$
  $onboard' = onboard \setminus \{p\}$

ENQUIRY OPERATIONS

- It would be useful to be able to determine how many people are on board the aircraft at any time. If this number is *numOnboard*, then

  $$numOnboard : \mathbb{N}$$
  $$numOnboard = \#\ onboard$$
  $$onboard' = onboard$$

  where the last statement clarifies that *onboard* does not change as a result of the query

- Another useful enquiry would tell whether a specific person is on board. Suppose the reply is a value of the *free type* RESPONSE, where:

  $$\text{RESPONSE} ::= \text{yes} \mid \text{no}$$

  Then, if $p : \text{PERSON}; \quad reply : \text{RESPONSE}$

  we have: $((p \in onboard \text{ AND } reply = \text{yes})$
  OR $(p \notin onboard \text{ AND } reply = \text{no}));$

  $$onboard' = onboard$$

FULL SYSTEM

- In summary the complete "state" description is:

[PERSON]	*The set of all possible uniquely identified persons*
$capacity : \mathbb{N}$	*The seating capacity of the aircraft*
$onboard : \mathbb{P}\ \mathrm{PERSON}$	*The set of persons on the aircraft (one of the many possible subsets of PERSON)*
$onboard \leq capacity$	*Constraint which is an invariant for system*
$onboard = \{\ \}$	*Initial state of the system*

Boarding is specified by:

$p : \mathrm{PERSON}$	*p is a person*
$p \notin onboard$	*Precondition for embarkation*
$\#onboard < capacity$	*Precondition for embarkation*
$onboard' = onboard \cup \{p\}$	*True after p embarks on aircraft*

Disembarkation is specified by:

$p : \mathrm{PERSON}$	*p is a person*
$p \in onboard$	*Precondition for disembarkation*
$onboard' = onboard \setminus \{p\}$	*True after p disembarks*

This page is intentionally blank

☺

EXERCISES

Scenario
A college provides a multi-user computer system for its students and staff. All staff and students must *register* with the college's IT Services unit before they are allowed access to the computer system. To use the system, each registered user must *log-in*. At any given time a registered user will either be *logged-in* or not *logged-in* (it is not possible for a user to be *logged-in* more than once concurrently).

Using the following declarations:

	[PERSON]	*the set of all uniquely identifiable persons*
users :	ℙ PERSON	*the set of all registered users*
logged_in :	ℙ PERSON	*the set of users who are currently logged-in*

express your solutions to the following symbolically (using sets) *and* in narrative form using **plain English.**

1. Discover any invariant properties of the system.

2. Define a suitable initial state for the system.

3. Define an operation to register a new user before their first *log-in*.

4. Define an operation to cancel a user's registration when the user is not *logged-in*.

5. Define operations for a registered user to *log-in* and to *log-out*.

(The example relating to 'passengers on an aircraft' should provide guidance)