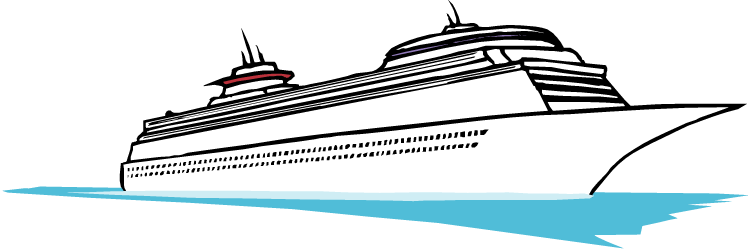
# 03. Boarding Passengers



*Embarking on a luxury cruise ship marks the beginning of a grand adventure, and efficient boarding is the gateway to unforgettable experiences at sea. Your function will serve as the captain of boarding, ensuring smooth sailing for all passengers from the moment they step on board. Your mission is to develop a Python function that orchestrates the boarding process seamlessly, balancing capacity constraints and efficiently managing passengers as they board the vessel.*

Write a function named **"boarding\_passengers"** that **receives information** about the available capacity of the ship and a passenger list and **returns the result after boarding**. The function will receive a **different number of arguments**. The arguments will be passed as follows:

* The **first** positional **argument** is the **capacity of the ship**:
  + An **integer** in the **range [0, 150]** inclusive.
* The second **group of arguments** represents the **passenger groups** asan **unknown number** **of tuples**:
  + Each **tuple** contains two elements:
    - the **number of passengers** as a **positive** **integer** in the **range [1, 150]** inclusive.
    - the **benefits program name** as a **string** (e.g., "**Diamond**", "**Platinum**", "**Gold**", "**First Cruiser**", etc.).

After receiving the information and calling the function, the program should **start the boarding process**:

* Board passengers **only** if you have **enough capacity**.
* Remember that you need to **track** the **total** **number of guests** based on their **benefit program**.
* If the **available capacity is not enough** to accommodate the **current group**, then **move to the next one** that can fit in.
* If the available **capacity** is **0** **(zero)**, **STOP boarding**!

To gather useful information, you need to **sort the boarding details**:

* Sort the information based on the boarded **number of guests** per each **benefit program** in **descending order**.
* If there is **more than one benefit program** with the **same number of guests**, order them according to their **benefits program name**, **ascending**.

**In the end, return** the output as described below.

***Note: Submit only the function in the judge system***

## Input

* There will be **no input from the console**, just parameters passed to your function

## Output

* Return the **sorted** **boarding details** per **each benefit plan**:  
  **"Boarding details by benefit plan:**  
  **## {benefit plan1}: {total number of passangers1} guests  
  ## {benefit plan2}: {total number of passangers2} guests  
  …  
  ## {benefit plann}: {total number of passangersn} guests"**
* Your **output string** should **also contain** one of the following messages:
  + If **all passengers are boarded**, return the message:   
    **"All passengers are successfully boarded!"**
  + If the **ship's capacity is occupied** but there are **still guests waiting to board**, then return the message:   
    **"Boarding unsuccessful. Cruise ship at full capacity."**
  + If there is **still available capacity** but **not all passengers have embarked the vessel**, return the message:   
    **"Partial boarding completed. Available capacity: {available\_capacity}."**

## Constraints

* The **first argument** will always be an **integer** in the **range [0, 150] inclusive**.
* The **second group of arguments** will be an **unknown number of tuples**.
* There will always be **at least one tuple**.

## Examples

|  |  |
| --- | --- |
| **Test Code** | **Output** |
| print(boarding\_passengers(**150, (35, 'Diamond'), (55, 'Platinum'), (35, 'Gold'), (25, 'First Cruiser')))** | **Boarding details by benefit plan:**  **## Platinum: 55 guests**  **## Diamond: 35 guests**  **## Gold: 35 guests**  **## First Cruiser: 25 guests**  **All passengers are successfully boarded!** |
| print(boarding\_passengers(100, (20, 'Diamond'), (15, 'Platinum'), (25, 'Gold'), (25, 'First Cruiser'), (15, 'Diamond'), (10, 'Gold'))) | Boarding details by benefit plan:  ## Diamond: 35 guests  ## First Cruiser: 25 guests  ## Gold: 25 guests  ## Platinum: 15 guests  Boarding unsuccessful. Cruise ship at full capacity. |
| print(boarding\_passengers(120, (30, 'Gold'), (20, 'Platinum'), (30, 'Diamond'), (10, 'First Cruiser'), (31, 'Platinum'), (20, 'Diamond'))) | Boarding details by benefit plan:  ## Diamond: 50 guests  ## Gold: 30 guests  ## Platinum: 20 guests  ## First Cruiser: 10 guests  Partial boarding completed. Available capacity: 10. |