COMP 636: Python Assessment

Due: **5pm Friday 29th March 2024**

Worth **40%** of COMP636 grade

Submit via Akoraka | Learn

Important note

This is an **individual** assessment. You must not collaborate or confer with others (e.g., telling others exactly what to do, or how to do it, or sharing or debugging others' code, or using ghost writers, etc), but the discussion of general concepts (e.g., how loops work **in general**, not specific to this assessment) is allowed. You may seek clarification and advice from staff.

DO NOT use any AI tools in the completion of this assignment, this includes but is not limited to: ChatGPT, Microsoft Co-Pilot, Google Bard.

Ensure you are familiar with the University policies on Academic Integrity (see here).

Introduction

Selwyn Campground has asked for a system to help manage its campground business. You are provided with an outline of the code for the system to complete.

The campground offers a range of options for people who wish to camp, these include unpowered tent-sites and powered sites.

Information is stored about each of the options offered, including the identifier and the maximum occupancy Each customer has contact information stored.

Example data is provided with this assessment and the format is described below.

UNPS is a list containing tuples that include an identifier for each Un-powered site and maximum occupancy for the site.

PS is a list containing tuples that include an identifier for each Powered site and a maximum occupancy for the site.

Bookings is a dictionary of campground bookings indexed by a key value (date), the entry is a list that contains a two lists, the first list is unpowered sites, the second list is for powered sites. Each list should contain tuples that have (site_identifier, customer_identifier, occupancy). Tuples are added to the list only when a booking is made.

An example of the bookings dictionary

```
{datetime.date(2024,4,11):[[('U01',563,1)],[('P04',652,3),('P07',732
,2)]]}
```

Customers is a dictionary of customers. The dictionary contains a dictionary per customer (key customer identifier (integer) customer details which is structured with the keys: name, phone and email (remember that keys don't need to be in the same order each time).

An example of the Customers dictionary is shown below.

```
{563:{'name':'Simon Smith',
'phone':'0244881901','email':'simon@smith.nz'}}
```

Assumptions: A site is listed in the bookings dictionary for a particular date only if it is booked on that date. A booking that covers multiple dates will be entered into the dictionary for each night of the booking. Customers are entered into the system before they can make a booking.

*Remember that we will use a different set of data for marking in camp_data.py with the same structure. So **do not hard-code values** because we will mark with different customers, sites and dates.*

File Download:

Download the following files from the COMP636 Assessment block on Akoraka | Learn:

- camp_admin_*your_name*.py – This is the initial code to begin from.
  Include your **own name** in the filename (e.g., camp_admin_**Anna_Lee**.py), and your name and student ID in a comment at the start of the file. **Do not change the menu numbering or existing function names**, although you may add additional functions of your own.
- camp_data.py – The campground data. **Do not change the structure** of this data. Although, you may add extra data. We will use our own copy of camp_data when marking, with different data, but with the same structure as provided.
- A function (next_id(dict)) is provided to give you the next ID number for a dictionary (add_customer)

Requirements

- The system needs to keep a record of customers.
- The system must allow bookings to be made by specifying a customer, a starting date, the number of nights (maximum five nights), and the booked site (powered or un-powered).

- A list of available camp sites needs to be displayed.
- Missing data values are recorded as None.

- One function (list_customers) for menu option 1 has already been provided for you. This lists all of the customers and displays Customer ID, Name, Phone Number and Email.

- You must use the provided **column_output** function for all on-screen display of data. You will need to convert your dictionary data into the correct format for this function (the list_customers function gives an example of how to do this). **Do not modify this function**.

- Validate all user input appropriately. If data of the wrong type is entered by the user, this should be captured without causing the program to crash or any other type of error. Also ensure that only valid values that make sense can be entered.

Tasks

Add the following features to the system:

1. **Menu enhancement:** Modify the code so that the user can enter an upper- *or* lower-case X (i.e., X or x) to exit the program.
2. **Add Customer:** Add a New Customer, taking in their name, phone number and email address. All values should be stored as Strings.
3. **List Camp Sites:** Show un-powered and powered sites, ordered by site. Information displayed should be the site identifier and maximum occupancy.
4. **Add Booking:** This function should add a booking for a customer, taking in the site to be used, the start night, and the number of nights. The interface must prompt the user with valid values to assist them entering information (e.g. list of site identifiers). The key value for a booking is the current date in the python date format (datetime.date(Year,Month,Day))

   Note: It is assumed that only one booking per day will be added per customer, you do not need to check/validate this.
5. **List campers for a date:** Display campers (customers) that are staying at the campground for a specified night. This should show customer name, site and number of occupants. The display should be ordered by site identifier.

Additional notes:

- The quality of the user experience will be taken into account, for each of the tasks above. Full marks for any item will require validation of data entered (for data type and sensible values) and details in the interface that demonstrate some consideration of what would work well for the user (within the limitations of the terminal window output in VS Code).
- The provided camp_admin_*your_name*.py Python file contains a menu structure and partially completed functions. These must not be deleted or renamed, but you may add arguments/parameters to these functions. You may also add additional functions of your own. Remember to rename the file to include your name.
- You are expected to apply problem solving skills to practically solve issues as they arise.
- You must add comments to your code. These do not need to be on every line of code but should be written to be enough detail so that if you came back to the code in 12 months' time that you could quickly work out what the code is doing. The existing list_customers function gives an example of the level of commenting that is expected. Marks for comments are included in the allocation for each menu item.

Submission:

Submit (upload) **only** your main Python .py file for marking: camp_admin_*your_name*.py. Do not include the camp_data.py file.

- Submit your file via the submission link on the COMP636 Assessment page.

Indicative mark allocation

(This *indicates* where to spend your time).)

40 marks available in total:

Item	**Approximate Marks available**
Menu enhancement	1
Add Customer	10
List Campsites	4
Add Booking	15
List Campers for a given date	10
TOTAL	**40**