NANYANG JUNIOR COLLEGE

JC2 COMMON TEST 2

Higher 2

COMPUTING 9569/02

Paper 2 (Lab-based) **6th July 2021**

3 Hours

Additional Materials: Removable storage device
Electronic version of surface_air_temperature.csv data file
Electronic version of school_info.csv data file
Electronic version of subjects_offered.csv data file
Electronic version of testvalues.txt data file
Insert Quick Reference Guide

READ THESE INSTRUCTIONS FIRST

Answer **all** questions.

All tasks must be done in the computer laboratory. You are not allowed to bring in or take out any pieces of work or materials on paper or electronic media or in any other form.

Approved calculators are allowed.

Save each task as it is completed.

The use of built-in functions, where appropriate, is allowed for this paper unless stated otherwise.

Note that up to 6 marks out of 100 will be awarded for the use of common coding standards for programming style.

The number of marks is given in brackets [ ] at the end of each question or part question. The total number of marks for this paper is 100.

This document consists of **10** printed pages.

NANYANG JUNIOR COLLEGE
Internal Examinations

© NYJC 2021

Instruction to candidates:

Your program code and output for each of Task 1 to 3 should be saved in a single .ipynb file. For example, your program code and output for Task 1 should be saved as

TASK1_<your name>_<class>.ipynb

1 Name your Jupyter Notebook as

TASK1_<your name>_<class>.ipynb

The task is to implement a histogram helper function. The function takes in an array of values (which may have duplicates), and returns a dictionary.

For each of the sub-tasks, add a comment statement, at the beginning of the code using the hash symbol '#', to indicate the sub-task the program code belongs to, for example:

```
In [1]:     #Task 1.1
            Program code

            Output:
```

Task 1.1

Write a function task1_1(in_array) that:

- takes an input array in_array
- creates a dictionary containing unique elements from in_array as keys, and the frequency (number of occurrences) of each key as the value
- returns that dictionary [5]

Test your function using the following **three** arrays (found in testvalues.txt):

- ["Alice", "Bob", "Alice", "Charlie", "Bob", "Bob", "Alice", "Charlie", "Alice"]
- ["1", "6", "10", "6", "2", "1", "9", "1", "6", "9", "1", "2"]
- ["Yes", "Yes", "No", "Yes", "No", "No", "No", "Yes", "No", "Yes"] [3]

Task 1.2

Such counting functions can be used to count the number of words in a sentence.

Write a function task1_2(sentence) that:

- takes in a string sentence representing a full sentence in English
- removes all punctuation
- splits the sentence into words
- uses the function in **Task 1.1** to count the number of occurrences of each word. [3]

Test your function with the following sentence (found in testvalues.txt):

- "All tasks must be done in the computer laboratory. You are not allowed to bring in or take out any pieces of work or materials on paper or electronic media or in any other form." [2]

Save your Jupyter Notebook for Task 1.

(Total: 13)

2 Name your Jupyter Notebook as

TASK2_<your name>_<class>.ipynb

The task is to:

- sort a list of currency denominations by size
- convert an integer amount into a collection of currency denominations

For each of the sub-tasks, add a comment statement, at the beginning of the code using the hash symbol '#', to indicate the sub-task the program code belongs to, for example:

```
In [1]:         #Task 2.1
                Program code

                Output:
```

Task 2.1

Write a function task2_1(list_of_integers) that:

- takes a list of integers, list_of_integers
- sorts them by size in descending order (largest to smallest) using insertion sort
- returns the sorted list [7]

Use the list [50, 10, 1000, 200, 500, 5000, 5, 100, 20] to test your function.

It should return the result [5000, 1000, 500, 200, 100, 50, 20, 10, 5]. [2]

Task 2.2

A digital cashier machine needs to dispense an amount of money to the customer with the least number of coins or notes.

The pseudocode for a function that takes in an integer Amt and an array of currency denominations Denom, and returns the shortest array of currency denominations that makes up Amt, is as follows:

```
FUNCTION GetChange(Amt: INTEGER, Denom: ARRAY) RETURNS ARRAY
  // Denom is an array of allowed denominations
  DECLARE Tray : ARRAY[0:] OF INTEGER  // Tray is an array of integers
  DECLARE i : INTEGER = 0  // Index of the Tray array
  DECLARE j : INTEGER = 0  // Index of the Denom array
  WHILE Amt > 0 AND j < Denom.LENGTH DO
    IF Amt >= Denom[j] THEN
      Tray[i] ← Denom[j]
      Amt = Amt - Denom[j]
      i = i + 1
    ELSE
      j = j + 1
    ENDIF
  ENDWHILE
  RETURN Tray
ENDFUNCTION
```

Write a function task2_2(amt, denom) that:

- takes in an integer amt representing an amount of money in cents, and an array of allowed currency denominations denom
- determine how many of each denomination is required to make up the amount of money
- return the breakdown as an array of values [5]

Test your function with the following input amounts, and the sorted denomination list from **Task 2.1**:

- 845 (should return [500, 200, 100, 20, 20, 5])
- 7840 (should return [5000, 1000, 1000, 500, 200, 100, 20, 20])
- 230 (should return [200, 20, 10]) [3]

(Total: 17)

3 Name your Jupyter Notebook as

TASK3_<your name>_<class>.ipynb

The text file surface_air_temperature.csv contains information on the monthly mean surface air temperatures in Singapore. Each line contains comma-delimited data on the daily minimum and daily maximum temperature for one month in the year.

Each line is in the following format:

YYYY-MM,<float>,<float>

- The date is in the form YYYY-MM, for example 1982-01 is the month of January in 1982
- the mean daily minimum and maximum temperatures are given in degrees Celsius with decimals.

The text file is stored in ascending order of date.

For each of the sub-tasks, add a comment statement, at the beginning of the code using the hash symbol '#', to indicate the sub-task the program code belongs to, for example:

```
In [1]:     #Task 3.1
            Program code

            Output:
```

Task 3.1

Write program code to:

- read in the surface air temperature data from a text file
- find the lowest daily minimum temperature **for each year**
- find the highest daily maximum temperature **for each year**
- print the year, the lowest daily minimum, and highest daily maximum temperature

Use surface_air_temperature.csv to test your program code. [9]

Task 3.2

The monthly temperature range is the difference between the lowest daily minimum temperature and the highest daily maximum temperature for that month.

Amend your program code **for each year** to calculate and display:

- the lowest daily minimum, and the (numerical) month in which it occurs
- the highest daily maximum, and the (numerical) month in which it occurs
- the largest monthly temperature range, and the (numerical) month in which it occurs

Use surface_air_temperature.csv to test your program code. [6]

Task 3.3

A researcher wants to analyse how the daily minimum and maximum temperatures have shifted over the past 20 years.

Amend your program code to:

- output the average daily minimum temperature of each month (Jan, Feb, ...) for the years 1982 to 2020 (i.e. average of 1982 Jan, 1983 Jan, 1984 Jan, ...)
- output the average daily maximum temperature of each month (Jan, Feb, ...) for the years 1982 to 2020 (i.e. average of 1982 Jan, 1983 Jan, 1984 Jan, ...)
- print, for each month **in 2020**, the deviation of daily minimum from the average (2020 minimum minus average minimum)
- print, for each month **in 2020**, the deviation of daily maximum from the average (2020 maximum minus average maximum) [6]

Your output format should be as follows:

```
Month 1: average daily min 1982-2020 was <float>, deviation in 2020 was <float> deg C.
    Month 1: average daily max 1982-2020 was <float>, deviation in 2020 was <float> deg C.
Month 2: average daily min 1982-2020 was <float>, deviation in 2020 was <float> deg C.
    Month 2: average daily max 1982-2020 was <float>, deviation in 2020 was <float> deg C.
...
Month 12: average daily min 1982-2020 was <float>, deviation in 2020 was <float> deg C.
    Month 12: average daily max 1982-2020 was <float>, deviation in 2020 was <float> deg C.
```

(Total: 21)

4A tuition centre uses a text file to store public data about schools in Singapore. Schools are located in a zone, and identified by which level of students they teach. The tuition centre wishes to migrate this information into a database. They also wish to incorporate information about the subjects offered by each school.

A web page will then be used to summarise the data. Different information will be visible on the web page, depending on the type of school displayed.

Task 4.1

Create an SQL file called TASK4_1_<your name>_<class>.sql to show the SQL code to create database schools.db with three tables: School, Subject, and SchoolSubject.

The School table will have the following fields:

- SchoolID – an integer value
- Name – the full name of the school
- Zone – the zone which the school is located in
- Level – the level of students that the school takes in: primary, secondary, mixed level, junior college, or centralised institute
- YearsOfStudy – the number of years a student takes to graduate

The Subject table will have the following fields:

- SubjectID – an integer value
- Name – the name of the subject

The primary key of the School table is the SchoolID column, while the primary key of the Subject table is the SubjectID column.

Save your SQL code as

TASK4_1_<your name>_<class>.sql [6]

Task 4.2

Python programming language and object-oriented programming will be used to publish the database content on a web page.

The class School will store the following data:

- id – stored as an integer
- name – stored as a string
- zone – stored as a string

The class has the following methods defined on it:

- safe_name() – returns a string containing no spaces, which may be safely used for older computer systems. This string should be constructed as follows:
  - name with all spaces replaced with underscore (_)
  - apostrophes ('), periods (.), and commas (,) removed
- as_record() – returns a tuple of values in the same order as the SQL table column order

Save your program code as

TASK4_2_<your name>_<class>.py [6]

The subclasses PrimarySchool, SecondarySchool, and JuniorCollege inherit from School, such that:

- they have *class attributes* with an appropriate value
  - the attribute level has values primary, secondary, or junior college
  - The PrimarySchool subclass has attribute YearsOfStudy = 6
  - The SecondarySchool subclass has attribute YearsOfStudy = 5
  - The JuniorCollege subclass has attribute YearsOfStudy = 2
- The as_record() method returns the appropriate values for the subclass

Add your program code to

TASK4_2_<your name>_<class>.py [3]

The text file, school_info.csv, contains data items for schools in Singapore. Each data item is separated by a comma, with each school's data on a new line as follows:

- numerical id
- school name
- zone (NORTH, SOUTH, EAST, or WEST)
- level (PRIMARY, SECONDARY, or JUNIOR COLLEGE)

Write program code to read in the information from the text file, school_info.csv, creating an instance of the appropriate class for each school. [5]

Write program code to insert all information from the file into the schools.db database.

Run the program.

Add your program code to

TASK4_2_<your name>_<class>.py [8]

Task 4.3

The text file, subjects_offered.csv, contains information about subjects offered by schools in Singapore. Each data item is separated by a comma, with data for each subject offering on a new line as follows:

- school ID
- school name
- subject offered

Write program code to read in the information from the text file, subjects_offered.csv, and insert all information from the file into the schools.db database.

Save your program code as

TASK4_3_<your name>_<class>.py [2]

Create an SQL file called TASK4_3_<your name>_<class>.sql to show a single SQL command used to look up the subjects offered by the school NANYANG JUNIOR COLLEGE.

Save your SQL code as

TASK4_3_<your name>_<class>.sql [5]

Task 4.4

The subjects offered by each junior college in Singapore are to be displayed in a web browser.

Write a Python program and the necessary files to create a web application that enables the subjects offered by NANYANG JUNIOR COLLEGE to be displayed.

The web page should include the:

- school name
- school zone
- list of subjects

Save your program as

TASK4_4_<your name>_<class>.py

with any additional files / sub-folders as needed in a folder named

TASK4_4_<your name>_<class> [5]

Run the web application and save the output of the program as

TASK4_4_<your name>_<class>.html [3]

(Total: 43)

----- END OF PAPER -----