JURONG PIONEER JUNIOR COLLEGE

2022 JC 2 Practical Revision Package 2

COMPUTING **9569/02**
Higher 2

Paper 2 (Lab-based) **3 hours**

1 The task is to read a single character from the keyboard, check that it is alphabetic. If it is, display the character, its denary, hexadecimal and binary value.

Task 1.1

Write program code to:

- prompt the user to enter a single letter
- read a character from the keyboard
- test that it is a single letter in the ranges `A-Z` or `a-z`.
- display the character if it is in the correct range
- or display a suitable error message and allow the user to input the character again. [4]

Evidence 1.1

Save your program as `TASK1_1.py`

Test your program with the following test data:

```
A
b
=
```

Place screenshots of your testing in your evidence document. [2]

Task 1.2

Write program code to:

- input a letter
- input a number base greater than `10` and less than `16`
- display the ASCII value of a single letter in denary
- convert the ASCII value to the number base input
- display the letter and both its values. [5]

Your screen display should look like this:

```
Letter                B
Denary                66
Number Base     11    60
```

Evidence 1.2

Save your program as `TASK1_2.py`

Test your program with the following test data:

```
B
m
```

Place screenshots of your testing in your evidence document. [2]

Task 1.3

Extend you program to display a menu showing the conversions available, and allow the user to choose the conversion. Your menu should look like this:

```
1.   Enter a letter
2.   Convert to Denary
3.   Convert to Base 11
4.   Convert to Base 12
5.   Convert to Base 13
6.   Convert to Base 14
7.   End
```

Place screenshots of your testing in your evidence document. [3]

Evidence 1.3

Save your program as `TASK1_3.py`

Test your program with the following test data:

```
X
Convert to Base 13
Convert to Base 11
y
Convert to denary
End
```

Place screenshots of your testing in your evidence document. [3]

2 The file `SONG.TXT` contains the lyrics of the song titled "*Count on me Singapore*".

The task is to read every word from the file, store it in a suitable data structure, sort the words in dictionary order (Lexicographical Order) and perform searches based on word and count.

Task 2.1

Write program code to:

- read the words from the file and store them in a suitable data structure,
- sort the words in dictionary order using **quick sort**,
- write each word and their number of occurrence in a text file, `WORDCOUNT.TXT`, where the next word is on a new line. [12]

A sample of the `WORDCOUNT.TXT` for the first **5 lines** is as follows:

```
a 5
achieve 12
air 1
all 1
and 9
```

Task 2.2

Write program code to:

- read the words and count from the `WORDCOUNT.TXT` file
- allow the user to select the following options:
  ```
  1. Search for a word
  2. Search for word(s) based on count
  3. Quit program
  ```
- take in user input for the word or the count
- if found, display the word(s) and the count, else display `"Not Found"`
- display appropriate error messages for invalid user input [6]

Task 2.3

Design test data for your program written in **Task 2.2**, provide evidence of testing that includes:

- search for a word that is contained in the file
- search for a word that is not contained in the file
- search for word(s) with a count that is contained in the file
- search for word(s) with a count that is not contained in the file [2]

Download your program code and output for Task 2 as
`Task2_<your name>_<centre number>_<index number>.ipynb`

3 An application is to be created to store a number of countries and their population as a direct access file. It is estimated there are around 250 countries.

This data is provided in file `COUNTRIES.txt`. Each country takes up one line, for example:
India 1173.108

Countries are written to and read from the direct access file using a hashing function. The address is calculated from a **hashing function** as follows:

- The ASCII code is calculated for each character within the country string
- The total of all ASCII values is calculated
- The total is divided by 373 and the remainder calculated with modulo arithmetic
- The value returned by the function is the (remainder + 1). This value is the address for the country name and population for this country.

For example, if the user inputs **India**, the value from the hashing function is 113. Therefore, write India to the file with address 113.

Task 3.1
Write program code for the hashing function using the following specification.

```
      FUNCTION HashKey (ThisCountry : STRING) : INTEGER
```

The function has a single parameter `ThisCountry` and returns the hash key (i.e. the address) as an integer.

Evidence 6: Your `HashKey` program code. [11]

Task 3.2
Write program code for a procedure `CreateCountry` which does the following:

- the program reads the **first** country name and the population from `COUNTRIES.txt`
- the address is hashed from the country name using function `HashKey`
- the data for this country is stored in the direct access file `NEWFILE`

Evidence 7: Your `CreateCountry` program code. [8]

Task 3.3

Write program code for a procedure `LookUpCountry` which does the following:

- the user inputs the country name
- the address is hashed from the country name
- the country and population is located in `NEWFILE`
- the address, the country name and population are output

Run the program and retrieve Afghanistan from `NEWFILE`

Evidence 8: Your `LookUpCountry` program code. [4]

Evidence 9: A screenshot confirming the retrieval of the Afghanistan data. [2]

Task 3.4

Amend your `CreateCountry` program code from Task 3.2 so that **all the country records** are read from `COUNTRIES.txt`, their addresses calculated and the country data is written to `NEWFILE`.

You must ensure that when a collision occurs your program design will deal with this situation with the result that all records are written to NEWFILE.

Add comment lines to your program code at the start of the procedure to describe your design for dealing with a collision.

Evidence 10: Your program code for the amended procedure `CreateCountry`. [9]

Task 3.5

Amend your `LookUpCountry` procedure to find the country data affected by collisions. You must ensure that the address output is the one where the country data has been stored.

Evidence 11: Your program code for the amended procedure `LookUpCountry`. [4]

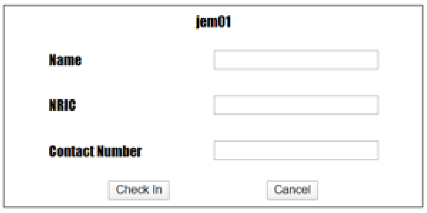
Evidence 12: Produce two screenshots showing the retrieval of **Chile** and **India** by the user.
[2]

4 In order to facilitate contact tracing, a web application is to be developed to allow visitors at different locations to check in when they entered the premises and check out when they leave the premises.

An example of the check in and out workflow is as follow:

(a) Visitor uses a mobile phone to scan a QR code encoded with the url, example http://localhost:5000/jem01.

(b) The following form appears on the visitor 's mobile web browser.



jem01

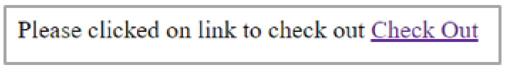
Name

NRIC

Contact Number

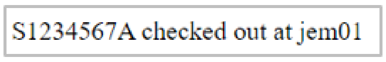
(c) Visitor fills up form and click Check In

(d) The web app will returned a web page containing a link for the user to check out when he/she leaves the premise as follows:



Please clicked on link to check out Check Out

(e) After visitor clicked on the Check Out link. The following message will be displayed on the mobile web browser:



S1234567A checked out at jem01

A database containing 2 tables with their data attributes is described as follows:

```
Location( LocationID:STRING, Name:STRING,
Address:STRING, URL:STRING)

Visitor (NRIC:STRING, LocationID*:STRING, Name:STRING,
Contact:STRING, Date:STRING, TimeIn:STRING, TimeOut:STRING)
```

Underline attribute : Primary Key
* Foreign Key

Task 4.1

Use the relevant tools or code to:

- Create a database named **`EntryDB`**.
- Create the 2 tables described above.
- Populate the `Location` table by importing the data from the file `LOCATIONS.CSV`
- If you are writing code, name your code file/s, `TASK4_1.py/TASK4_1.sql` [2]

At the entrance of each location, a QR code is encoded with the URL of the location that is stored in the `Location` table in Task 4.1.

When a visitor entered the premise, he/she will scan the QR code and the web browser will be directed to the URL for that location. The format of the URL is as follows: `http://<server_dns_name>/<location_id>.`

where

- `<server_dns_name>` is the dns name of the web server.
- `<location_id>` is the value of the LocationID attribute stored in the location table.

A check-in form will be rendered on the web browser for the visitor to entered the information required to create a record in the Visitor table. The visitor will only need to enter.

- `name,`
- `NRIC,`
- `contact number.`

Task 4.2

- Create a jinja template named checkin.html
    - The LocationID must be auto-filled and displayed in the form.
- Write Python code to rendered the checkin.html form when the visitor checks in by scanning the QR code on his/her mobile device.
- You can assume that the QR code scanner app will automatically launch a web browser and fills in the URL on the address bar.

[5]

Task 4.3

Write Python code to

- retrieve the data in the form submitted by the visitor and insert a new Visitor record in the Visitor table. In addition, the following attributes in the table are to be generated automatically:
    - Date in YYMMDD format.
    - Time In in HHMM 24 hour.
      (YY: Year, MM:Month, DD:Day, HH:Hour, MM:Minute)
- Return a web page containing a link for the visitor to check-out as described in the example workflow. [5]

Task 4.4

When the visitor clicked on the Check Out link as described in the example workflow, write Python code to:

- Update the Timeout in the visitor's record to indicate the time that the visitor leaves the premise
- Return a web page with a message with the visitor's NRIC as described in the example workflow. [3]

Save your Python program as

`TASK4_4_<your name>_<NRIC number>.py`

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

`Task4_<your name>_<NRIC number>`