FIT1008 – Intro to Computer Science
Assessed Prac 1 – Weeks 3 and 4

Semester 2, 2017

Objectives of this practical session

To be able to write MIPS programs involving lists, local variables, and functions.
Note:

- **Local variables must be stored on the runtime stack.**
- **For all the MIPS programs you should first implement a Python version, and work out a faithful translation into MIPS. Both, the Python and MIPS code are required for your prac to be marked. However, Python solutions alone do not result in any marks, all exercises are about MIPS and MIPS solutions are required in all exercises to achieve more than 0 marks.**
- **Use only instructions on the MIPS reference sheet and use comments to document each piece of code.**
- **You may copy and paste your code between tasks where you need to reuse code.**
- **Create a new file/module for each task or subtask.**
- **Name your files task[num]_[part] to keep them organised.**

Task 1 [10 marks]

In the Gregorian calendar a *year* is a *leap year* if the *year* is divisible by 4 but not divisible by 100, or if the *year* is divisible by 400.

(a) Write a Python program `task1_a.py`, which reads in a year (i.e., an integer ≥ 1582), and if the *year* is a leap year prints "Is a leap year", otherwise prints "Is not a leap year". In your module you should include a function with the signature `def is_leap_year(year)` that returns true if it is a leap year otherwise it returns false. You should avoid doing any input or output in this function, instead do the input and output as part of the **main** function..

Run `test_task1_a.py` to ensure it works properly. Alternatively you can write your own test.

(b) Re-write your Python program without using any user-defined functions. This will help you with a faithful MIPS translation in the following sub-question.

(c) Write a MIPS program which implements `task1_b.py`.

Task 2 [10 marks]

(a) Write a Python program `task2_a.py`, which performs the steps below.

- Reads in the size of the `the_list`.
- Reads in all the items of `the_list`.
- Prints out every other element in `the_list`. That is, it always prints the first element, skips the second, prints the third, skips the fourth and so on.

Replace any Python **for** loops for **while** loops in order to facilitate your MIPS translation.

(b) Write a MIPS program which implements `task2_a.py` faithfully – there is no need to use functions at this stage.

Task 3 [20 marks]

(a) Write a Python program `task3_a.py`, which does the following:

- Reads in the size of the `the_list`.
- Reads in all the items of `the_list`, storing them in a list.
- Prints the range of the list. The range is defined as the difference between the maximum and the minimum element in the list.

Replace any Python **for** loops for **while** loops in order to facilitate your MIPS translation.

(b) Write a Python program (`task3_b.py`) which implements `task3_a.py` without using any defined functions. Again, this is in preparation for the following MIPS translation.

(c) Write a MIPS program which implements `task3_b.py` faithfully.

Background

The bureau of climate research has a device that records the average temperature in the city for each one of the days in a month. The office is interested in understanding climate variation and has commissioned you to design a collection of programs to help analyse the data collected. The programs will run in a small portable device based on a MIPS processor. For each task, first design the algorithm to be used in Python then write the algorithms in MIPS. In each task, the input list is a list containing a number for each day in the month, one temperature record per day.
An example of one such list is:
`26, 18, 22, 20, 13, 22, 19, 22, 20, 27, 18, 24, 15, 28, 26, 27, 20, 21, 23, 24, 27, 26, 15, 23, 22, 20, 23, 17, 18, 18`
This list is to be read in at the beginning of each task. We assume that the list will only contain natural numbers. The result of each task must be printed. The following tasks describe the functionality required.

Task 4 [20 marks]

Write some code to find how many times a given temperature appears in the list. The user should be able to enter the list (including its size), and then the target number that he/she wishes to count.

CHECKPOINT

(You should reach this point during week 3)

Task 5 [10 marks]

Write a function that finds the minimum temperature in the list. And use the function to write a program that prints the minimum temperature in any list given by the user. **Suggestion:** Write your program without using functions first.

Task 6 [20 marks]

Write a function that prints the frequency of each temperature in the list between two given temperatures a and b (such that $a < b$). It is a good idea to decompose this problem in several functions using Python, and then do a faithful translation. Make sure you use Python structures that you know how to translate into MIPS. For example, use lists instead of dictionaries.
For example, if the input is the list given in the **Background** section, and the user also gives $a = 20$ and $b = 26$, the output should be:
20 appears 4 times
21 appears 1 times
22 appears 4 times
23 appears 3 times
24 appears 2 times
26 appears 3 times
You are encouraged to use re-use functions you have written for other tasks in this assignment.

Task 7 [10 marks]

Now we want to store a range of temperatures for different cities. The temperature data should be stored in a matrix, such that each row represents a city and each column represents a day's temperature. Write a program to read in the number of days and the number of cities. The program should then use a menu to ask the user for a city and print the maximum temperature in such city.
Note: Cities are represented by numbers, so there is no need to ask, or store, for any string, simply numbers.