THE UNIVERSITY OF WESTERN AUSTRALIA

DESK No. | | | |

FAMILY NAME: _____

GIVEN NAMES: _____

SIGNATURE: _____

STUDENT NUMBER: | | | | | | | | |

SEMESTER 2, 2022 EXAMINATIONS

CITS2002

Systems Programming

This paper contains: **6** Pages **(including title page)**

Time Allowed: **2:00** hours

INSTRUCTIONS:

This is a CLOSED BOOK examination.

You may bring to the examination and use 1 page (2 sides) of handwritten or typed notes.

The paper contains 6 pages and 5 questions.

Each question is worth 10 marks.

You are required to attempt ALL 5 questions

THIS IS A CLOSED BOOK EXAMINATION

SUPPLIED STATIONERY	**ALLOWABLE ITEMS**
1 x Answer Booklet 10 pages	**1 page (2 sides) of handwritten or typed notes.**

PLEASE NOTE

Examination candidates may only bring authorised materials into the examination room. If a supervisor finds, during the examination, that you have unauthorised material, in whatever form, in the vicinity of your desk or on your person, whether in the examination room or the toilets or en route to/from the toilets, the matter will be reported to the head of school and disciplinary action will normally be taken against you. This action may result in your being deprived of any credit for this examination or even, in some cases, for the whole unit. This will apply regardless of whether the material has been used at the time it is found.

Therefore, any candidate who has brought any unauthorised material whatsoever into the examination room should declare it to the supervisor immediately. Candidates who are uncertain whether any material is authorised should ask the supervisor for clarification.

Candidates must comply with the Examination Rules of the University and with the directions of supervisors.

No electronic devices are permitted during the examination.

All question papers and answer booklets are the property of the University and remain so at all times.

This page has been left intentionally blank

1) A company makes monthly backups of its data into a directory named */backups*, separating data for the company's departments into subdirectories, with names such as */backups/products*, */backups/sales*, and */backups/logistics*, and with each subdirectory possibly containing multiple subdirectories (and so on).

Unfortunately, the company has no consistent naming scheme for its files and directories, and simply uses each file's modification time to record the month for which its data applies.

The amount of data backed-up each month is very seasonal, with the amount of data backed-up in, say May 2021, usually a good predictor of the amount of space required for May 2022. Unfortunately, COVID-19 lockdowns over the past 2 years, have resulted in the company shutting down for several months and, thus, some months have no backed-up data.

Write a C99 program which accepts a single command-line argument representing a month, with *0* representing January, *1* representing February, and so on, to *11* representing December.

After recursively examining the */backups* directory, the program should simply print the total size, in bytes, used to store the *most recently backed-up data* from the requested month.

For example, if the program is run in May 2022 with a command line-argument of *7* (representing August), the program should simply print the total number of bytes used to store the data of *the most recent August* that has been backed-up.

If no backups were found for August 2021, the program may report on the backups of August 2020. If nothing is found for August 2020, then August 2019 may be the most recent August, and so on.

(10)

---

2) An author writes each chapter of their books in separate text files. To quickly find the places where each person's name appears in the books, the author always writes each name entirely in uppercase, for example MARCIA, JAN, CINDY, GREG, PETER, and BOBBY.

Names appear as space-separated words in each text file, where a word consists entirely of alphabetic characters, after all non-alphabetic characters have been replaced by spaces.

The author simply names each book after the book's most frequently mentioned person.

Write a C99 program which accepts the names of one-or-more text files as command-line arguments, typically the chapters of a book, and simply prints the name of the most frequently mentioned person.

(10)

---

3) Consider the following C99 function prototype:

```
int compileSourceToObject(char *sourcefiles[]);
```

The function receives a NULL terminated array of character strings, providing the names of C99 source code files which, together, form a large multi-file project.

The `compileSourceToObject()` function attempts to compile each source code file to produce its corresponding object file, but *does not* link all of the object files together.

The function returns 0 if *all* source code files can be successfully compiled to object files, and returns 1 otherwise.

Write the `compileSourceToObject()` function in C99.

You may not use the standard POSIX function `system()`, and should assume that the C99 compiler is located at */usr/bin/cc*

(10)

4a) Explain clearly the following state transitions for processes and the reasons for the transitions:

1. from Running to Blocked.

2. from Blocked to Blocked-Suspend.

(5)

4b) Explain the importance of logical to physical address translation.

Consider a computer in which each 18-bit logical address has a 12-bit page offset. Explain how the logical to physical address translation is performed for this computer.

Independent of other operating system imposed constraints, what is the maximum number of pages that a process can be allocated in this computer?

(5)

---

5) As operating systems evolve, the number of system-calls that they support grows.

However, with very few exceptions, each new system-call follows a fairly standard 'pattern' in how it is called from application programs, how it executes, and how it returns its results.

With reference to a significant example involving a calling sequence of several related system-calls, explain the decisions that have been taken in the design of the interface between application programs and system-calls.

In your answer make reference to how system-calls are referenced in application programs, how parameters are passed, how results are returned, and how the application program is informed of any errors that may have occurred with the call.

(10)

END OF PAPER