University of Glasgow

Tuesday 7 February 2023
09:30-11:30 GMT
Duration: 1 hour 30 minutes
Additional time: 30 minutes
Timed exam — fixed start time

DEGREE of BSc Software Engineering (Graduate Apprenticeship)

SYSTEMS PROGRAMMING
COMPSCI 2030

(Answer all 3 questions)

This examination paper is an open book, online assessment and is worth a total of 60 marks.

1. Systems Programming in C

This question requires long-form answers. There are **30** marks available in total in this question.

(a) Consider the following declaration: `int matrix[40][20]`
Express the address of the first element in the 20th row in **two** different ways. Note that arrays start at index 0. [4]

(b) What is the value of `*(p+1)` after executing the following code fragment?
Please give a number.

```
int *p;
int sims[3][3] = {1, 2, 3, 4, 5, 6, 7};
p = sims[1];
```

[2]

(c) What data type would be the most suitable to use for each of the following kinds of data?
In each case, pick **exactly one** from the following data types: **`char`**, **`int`**, **`float`**, **`double`**.

(i) The number of undergraduate students in the School of Computing Science. [1]

(ii) The value of π to 10 decimal places [1]

(iii) The cost of a new smartphone [1]

(iv) A key that a user presses to take an action in a text-based adventure game [1]

(d) Declare a function called `sub_str` that returns a pointer to a string, and takes as arguments a string and an integer. [4]

(e) Consider the following table of data types and the corresponding bytes allocated for each in memory.

Data Type	Size
char	1 byte
int	4 bytes
float	4 bytes
double	8 bytes
pointer	8 bytes

Consider also the following **`struct`** definition and use.

```
struct house {
    char* name;
    char address[40];
    int num_of_floors;
    int num_of_rooms;
    int num_of_bathrooms;
    float total_area;
};
struct house home = {"White House", "308 Negra Arroyo Lane", 2, 8, 1, 1910.25};
```

(i) How many bytes of **stack memory** will the data stored in the `home` variable occupy? Explain how you calculated your answer. [2]

(ii) How many bytes of **heap memory** will the data stored in the `home` variable occupy? You can assume that appropriate `malloc` call(s) were made where necessary. Explain how you calculated your answer. [2]

(f) The size of a pointer depends on the data type it points to. True or false? [1]

(g) The size of a pointer depends on the architecture of the machine. True or false? [1]

(h) The program below takes two command-line arguments: a character and a filename, respectively. The program should print all the lines in the file that contain the given character at least once. Write code for the missing `has_ch` function.

```c
#include<stdio.h>
#include<stdlib.h>
#define BUF 256

int main(int argc, char * argv[]) {
    FILE * fp;
    char ch;
    char line[BUF];

    if (argc!=3) {
        printf("Usage: %s character filename\n", argv[0]);
        exit(EXIT_FAILURE);
    }
    ch = argv[1][0];
    if ((fp=fopen(argv[2], "r"))==NULL) {
        printf("Error: Cannot open %s\n", argv[2]);
        exit(EXIT_FAILURE);
    }
    while (fgets(line,BUF,fp) != NULL) {
        if (has_ch(ch,line)) {
            fputs(line, stdout);
        }
    }
    fclose(fp);
    return 0;
}
```

[10]

2. Concurrent Systems Programming – Multiple-Choice Questions

These multiple-choice questions are negatively marked. For each correct answer you gain two marks, and for each wrong answer, you lose one. There is one correct answer per question.

There are **10** marks available in total in this question.

(a) Which of the following statements about processes is *not* true?

1. A process is a program in execution
2. Many threads can exist within a process
3. There is one program counter per process
4. Part of the state of a process is a list of open files

[2]

(b) Which of the following statements about threads and processes is true?

1. All threads within a process share heap space
2. All processes within a thread share heap space
3. Child processes cannot outlive their parent processes
4. When a thread exits, it has to close the process/processes associated with it

[2]

(c) Which of the following statements about mutexes is true?

1. Mutexes are to prevent user-after-free errors
2. Mutexes are used to protect critical sections
3. Each thread using a mutex needs its own instance of said mutex
4. Mutexes allow many threads to access a resource simultaneously

[2]

(d) When each of two threads is waiting on a resource that is being held by the other thread, this is typically referred to as a

1. Lifelock
2. Deadlock
3. Starvation
4. Mutual exclusion

[2]

(e) Amdahl's law describes:

1. The maximum speedup we can achieve not protecting critical sections
2. The speed of a program which does not make use of parallelism
3. The maximum speedup of a program that can be achieved by parallel execution

4. The amount of additional work we can do with more parallel execution units for a given algorithm

[2]

3. Concurrent Systems Programming – Long-Form Answer Questions

This question requires long-form answers. There are **20** marks available in total in this question.

(a) Why would you use multiple *processes* to implement one application? Name two reasons and explain each briefly. A sentence or two should suffice for explaining each.

[4]

(b) The cooperating threads T1 and T2 are executed in parallel. Their code is given below. Indicate the different possible execution sequences and what values the two variables (`x` and `y`) assume after each, when `x` is initially set to `2` and `y` to `4` and each line is an atomic instruction.

```
1 T1:
2   x = x + x + 4;
3   y = x + y;
4
5 T2:
6   x = x + 4;
```

[4]

(c) A programmer has written the following C code that is intended to count how many times a function is called:

```
1 int total_calls = 0;
2
3 int count_calls() {
4     int calls_before = total_calls;
5     total_calls = calls_before + 1;
6     return total_calls;
7 }
```

This works well until the programmer creates multiple threads and has them repeatedly call this function. The programmer then finds that the number of calls is counted wrong.

Answer the following three questions briefly:

(i) Referencing particular lines of code, what can happen when multiple threads execute the `count_calls` function concurrently that leads to a wrong count?

[2]

(ii) By the end of the program, which has multiple threads repeatedly call the `count_calls` function, will the total number of calls counted be more or less than the actual number of calls, if they are not equal?

[1]

(iii) Which line or lines of code are problematic here and why?

[2]

(d) Explain what mutual exclusion means and which functions of the `pthread` library can be used to realize it using a mutex lock.

[2]

(e) Briefly give two advantages of using virtual memory addresses and address translation.

[2]

(f) In the context of computer memory/storage briefly explain what caching means.

[1]

(g) Consider the following levels of memory/storage: caches, disk, registers, main memory. Name which memory/storage level is usually the fastest (lowest access latency) and which is the largest (greatest storage capacity).

[2]

END OF QUESTION PAPER