A38279

Calculators may be used in this examination provided they are not capable of being used to store alphabetical information other than hexadecimal numbers



UNIVERSITY OF BIRMINGHAM

School of Computer Science

Second Year Undergraduate

06-38059

LI Operating Systems and Systems Programming

Main Summer Examinations 2025

Time allowed: 2 hours

[Answer all questions]

Note

Answer ALL questions.

Question 1

The following C functions (partially) implement adding and removing nodes from a binary tree:

```
01 #include <stdlib.h>
02
03 typedef struct treeNode {
04  int value;
05  struct treeNode *left;
06  struct treeNode *right;
07 } treeNode_t;
08
09 // Alloc the two nodes under a given one
10 void allocNodes(treeNode_t *node) {
11  node->left = calloc(1, sizeof(treeNode_t));
12  node->right = calloc(1, sizeof(treeNode_t));
13 }
14
15 // Deletes a node and the subtree under it
16 // Should release all memory
17 void deleteNode(treeNode_t *node) {
18  // free mem if not null
19  if(node != 0) {
20      free(node->value);
21      free(node);
22  }
23
24  // free left and right nodes
25  deleteNode(node->left);
26  deleteNode(node->right);
27 }
28
29 void doTree() {
30  treeNode_t *root = calloc(1, sizeof(treeNode_t));
31  allocNodes(root);
32  allocNodes(root->left);
33  deleteNode(root);
34 }
```

(a) When the function `doTree()` in the above code is called, the program crashes. Point to all line(s) that can cause this, and briefly explain why. **[8 marks]**

(b) Fix the problem(s) from (a) in the above code. You do not need to write down the full code, just the function(s) where you made changes. **[12 marks]**

(c) Explain very briefly why undefined behaviour in C is dangerous and must be avoided. **[6 marks]**

(d) Are the `allocNodes()` and `deleteNode()` functions thread-safe? If your answer is "yes", provide a brief justification. If you answer is "no", briefly explain which changes would be needed to make these functions thread-safe. **[10 marks]**

Question 2

The following C code shows two functions, `producer()` and `consumer()`, that run in parallel as separate threads, managed via the `pthread` API:

```
01 #include <pthread.h>
02
03 unsigned int request_count = 0;
04 unsigned int requests[8] = { 0 };
05
06 void producer() {
07   if(request_count < 8)
08   {
09     // add request into queue
10     requests[request_count] = get();
11     request_count++;
12   }
13 }
14
15 void consumer() {
16   if(request_count > 0) {
17     // take request from queue and handle
18     request_count--;
19     handle(requests[request_count]);
20   }
21 }
```

(a) Explain which concurrency problem(s) exists with the above code. Assume that the functions `get()` and `handle()` are thread-safe functions that get and process request data, respectively. Moreover, assume that whenever `get()` is called, the resulting request must be inserted into the `requests` array. **[9 marks]**

(b) Give the code for `producer()` and `consumer()` that resolves the problem from (a). Assume that you have a mutex declared as `pthread_mutex_t m`, with the respective functions from the lecture operating on this mutex being:

```
int pthread_mutex_lock(pthread_mutex_t *);
int pthread_mutex_trylock(pthread_mutex_t *);
int pthread_mutex_unlock(pthread_mutex_t *);
```

Your solution should maximise the degree of possible parallelism. **[9 marks]**

(c) Is it true that every program has at least one thread, even if one is not created explicitly? Justify your answer. **[6 marks]**

(d) Someone states that system calls are only necessary when accessing privileged resources like kernel memory or hardware, and that reading/writing from/to the console using `scanf()`/`printf()` does not require a system call. Is this true? Justify your answer. **[6 marks]**

Question 3

Consider a system running the following programs:

- A program for verifying properties of programs automatically, which is CPU-intensive (it may even run for hours) and uses large amounts of memory.
- A model compiler, which translates a text file containing the description of the properties into a format which the verification program understands. This compiler uses a library which is stored on a different server. This program is computationally intensive, uses a lot of I/O but is not memory intensive.
- A specialised editor for editing the description file. This program is not CPU-intensive and uses little memory.

The editor should be responsive at all times.

(a) Is there a scheduling strategy for this scenario which does not use pre-emption? Justify your answer. **[4 marks]**

(b) Can the interrupt handler which processes the packets from the library server store the received data directly into the memory area of the compiler process? Justify your answer. **[6 marks]**

(c) Describe a suitable scheduling strategy for this scenario and justify why your chosen strategy is suitable for this scenario. **[12 marks]**

(d) Describe a suitable memory management strategy for this scenario and justify why your chosen strategy is suitable for this scenario. **[12 marks]**

A38279 LI Operating Systems and Systems Programming Non-alpha only

Do not complete the attendance slip, fill in the front of the answer book or turn over the question paper until you are told to do so

Important Reminders

- Coats/outwear should be placed in the designated area.
- Unauthorised materials (e.g. notes or Tippex) must be placed in the designated area.
- Check that you do not have any unauthorised materials with you (e.g. in your pockets, pencil case).
- Mobile phones and smart watches **must** be switched off and placed in the designated area or under your desk. They must not be left on your person or in your pockets.
- You are not permitted to use a mobile phone as a clock. If you have difficulty seeing a clock, please alert an Invigilator.
- You are not permitted to have writing on your hand, arm or other body part.
- Check that you do not have writing on your hand, arm or other body part – if you do, you must inform an Invigilator immediately
- Alert an Invigilator immediately if you find any unauthorised item upon you during the examination.

Any students found with non-permitted items upon their person during the examination, or who fail to comply with Examination rules may be subject to Student Conduct procedures.