DUBLIN INSTITUTE OF TECHNOLOGY

DT228 BSc. (Honours) Degree in Computer Science

Year 2

DT282 BSc. (Honours) Degree in Computer Science (International)

Year 2

WINTER EXAMINATIONS 2017/2018

OPERATING SYSTEMS 2 [CMPU2017]

MR. DENIS MANLEY
DR. DEIRDRE LILLIS
MR. PATRICK CLARKE

WEDNESDAY 10TH JANUARY 2.00 P.M. – 4.00 P.M.

TWO HOURS

ANSWER QUESTION 1 AND ANY TWO OTHERS.

QUESTION 1 IS WORTH 40 MARKS, ALL THE REST ARE WORTH 30.

1

a) The race problem occurs if two or more concurrent process are not carefully synchronised. Explain, using an example, what happens if the race problem occurs.

(6 marks)

b) Given the following arrival times and CPU time for 4 processes determine the average turnaround time for a round robin schedule algorithm with a time slice of *4 ms.*

(8 marks)

Arrival Time	0	1	7	[illegible]
Job	A	B	C	D
CUP cycle time	8	4	9	5

c) Explain, using suitable examples, the steps required to push a node onto a stack.

(6 marks)

d) In order to implement a linked list using functions a pointer to the first node in the list must be passed by reference. Explain, using a suitable example, how you would pass a pointer variable by reference. **(5 marks)**

e) The following is sample code is to be used to insert a node to a link list; clearly explain what the code is doing, and if it will add a node to the list; assume that the link list is initially empty. **(15 marks)**

```
// link list add node (warning and not add node to list
void insert(ListNode* *sPtr, char value)
{
  ListNode* newPtr = malloc(sizeof(ListNode)); // what code does here

  //what the code does here
 if (newPtr != NULL) { // is space available
    newPtr->data = value; // place value in node
    newPtr->nextPtr = NULL; // node does not link to another node

    ListNode* previousPtr = NULL;
    ListNode* currentPtr = *sPtr;

    // what the code does here
    while (currentPtr != NULL && value > currentPtr->data) {
      previousPtr = currentPtr;  ...
      currentPtr = currentPtr->nextPtr;
    }

    // What is the code doing here.
    if (previousPtr == NULL) {
      newPtr->nextPtr = sPtr;
      sPtr = newPtr;
    }
    else { // What the code does here
      previousPtr->nextPtr = newPtr;
      newPtr->nextPtr = currentPtr;
    }
  }
  else {
    printf("%c not inserted. No memory available.\n", value);
  }
}
```

2
a) Distinguish between a single and a multi-threading process. **(4 marks)**

b) In C a thread is created using the following code:

int pthread_create(*pthread_t *tidp, pthread_attr_t *attr, *start_rtn, void * arg*)

Clearly explain what each of the arguments in the thread create function mean.
(8 marks)

c) Explain in your own words the following code: **(10 marks)**

```
#include<pthread.h>
#include <stdio.h>
int value;
void *my_thread(void *param);

int main (int argc, char *argv[])
{
pthread_t tid; int retcode;

//what does this do
if (argc != 3) {
    fprintf (stderr, "usage: a.out <integer value>\n");
    exit(0);
}

//what does this do
retcode = pthread_create(&tid,NULL,my_thread,argv[2]);

//what does this do
if (retcode != 0) {
            fprintf (stderr, "Unable to create thread\n");
            exit (1);
    }

// What does this do
    pthread_join(tid,NULL);
    printf ("I am the parent: the cube of value passed = %d\n", value);
} //main
```

```
// What does this do
void *my_thread(void *param)
    •   {
        int i = atoi (param);
        printf ("I am the child, I am passed value %d\n", i);
        value = i * i;*i
        pthread_exit(0);
    •   }
```

d) What would be the output of the executable code of above program. Explain your explanation should assume the following is input at the command prompt: **(4 marks)**

```
../a.out  text.txt
./a.out  text.txt 5
```

e) What would be the two outcomes in the above program if the *pthread_join* command was removed and the command line input was *./a.out text.txt 6*? Explain the reason for your answer. **(4 marks)**

3

a) Briefly describe the elements of a process control block (PCB) **(4 marks)**

b) In Linux a process is created using the *fork()* command. Explain, using an example the purpose of the fork command and how it achieves this purpose. **(6 marks)**

c) Explain, using a suitable example, the purpose of the *wait()* function. **(8 marks)**

d) The execvp linux command has the following function prototype format

*int execvp(char *prog, char * argv[])*

Explain, using an example, what this function will do and what would data be stored in the prog parameter and the arg parameter. **(4 marks)**

e) The following code uses fork(), wait() and exec() functions. Explain, using an example, what the code does at the places indicated. argv contains the values cp file1 and file2 where cp is the linux copy function. **(8 marks)**

```
•   void function1(char **argv)
•
        pid_t pid // a pid data type
        int status

        if ((pid = fork()) < 0) {                      /*what happens here    */
            printf("*** ERROR: forking child process failed\n");
            exit(42);
        }
        else if (pid == 0) {


            if (execvp(*argv, argv) < 0)               /* what happens here.*/
                printf("*** ERROR: exec failed\n");
                exit(1);
            }
        }
        else {
            while (wait(&status) != pid)              /* what happens here.  */
```

;

}

4:

a) Explain, using an example how you would map the logical address of a process to its physical address **(6 marks)**

b) What do you understand by the word trashing as it applies to virtual memory? **(3 Marks)**

c) Describe the purpose of each of the following field in the page map table of a virtual; memory system:

modified field

page frame field. **(4 marks)**

d) Page swapping is an essential element of virtual memory: two page swapping algorithms are the *First In First Out (FIFO)* and *Least Recently Used (LRU)* algorithms. Using a demand page system with 3 frames how many page faults will be generated by the following sequence? Clearly show how you arrived at the answer.

Reference Sequence = [E, G, A, B, E, F, A, C, B, D, E, A, F].

(12 marks)

e) A *cache* improves the speed of processor access to instructions and data. Explain how the cache improves the speed of virtual memory page swapping.

(5 marks)

COLLEGE EXAMINATIONS

AMENDMENTS TO EXAMINATION QUESTION PAPER

COURSE REF CMPU 2017 VENUE: Aungier St. Courtyard.

SUBJECT: Operating Systems 2

DATE: 10/1/18

TIME: 15:05

SIGNED: [illegible]

INSTRUCTIONS: Q 2(c)

value = i * i * i;

ie semi colon should be at the end of the statement.

COLLEGE EXAMINATIONS

AMENDMENTS TO EXAMINATION QUESTION PAPER

COURSE REF Prog 9810 VENUE: 943

SUBJECT: Programming Paradigms

DATE: 10/1/18

TIME: 15.45

SIGNED: Nuala [illegible]

INSTRUCTIONS: Question 4

second part "a" should be "b".

i.e. b. The following is an Haskell...

QUESTION 4 Functional Programming [33 marks]

a. Define an Haskell function `c_digit` that counts the number of occurrences of letters representing digits (i.e. '0','1','2','3','4','5','6','7','8','9') in a string. Include the function signature. Example:

```
c_digit "I am 12. I live in Dublin 2".

[('1',1),('2',2),('3',0),('4',0),('5',0),('6',0),('7',0),('8
',0),('9',0),('0',0)]
```

[11 marks]

b. ~~X~~ The following is an Haskell implementation of a famous sorting algorithm.

(i) Can you identify the algorithm? Can you explain its functioning by commenting each line of code and providing an example?

```
f []      = []
f (x:xs)  = f ys ++ [x] ++ f zs
            where
               ys = [a | a ← xs, a ≤ x]
               zs = [b | b ← xs, b > x]
```

[7 marks]

(ii) Provide a signature for the function *f*?

[3 marks]

[10 marks in total]

(iii) What is the output of the following command in Haskell?

```
(i)   :t ['a','g']
(ii)  :t True
(iii) :t ["HELLO!"]
(iv)  :t (True, 'a')
(v)   :t 4 == 5
```

[5 marks]

(iv) Describe the difference between an eager evaluation and a lazy evaluation of functions. Compare the advantages and disadvantages of each method.

[7 marks]