W228/205C

DUBLIN INSTITUTE OF TECHNOLOGY

BSc. (Honours) Degree in Computer Science

Year 2

WINTER EXAMINATIONS 2014/2015

Operating Systems 2 [CMPU2017]

Dr John Gilligan
Dr Deirdre Lillis
Mr. Kevin Foley

Monday 05 Jan 1.00 p.m. - 3.00 p.m.

Answer Questions 1 and any two others

Question 1 is worth 30 marks, all the rest are worth 35.

1: a) What is meant by the pointer construct in C and what briefly are their advantages?

(8 Marks)

b) Describe with the aid of a suitable example, how pointers are used to pass data by reference to functions in C.

(8 Marks)

c) Write a program in C which accepts in an array of 10 Integers and uses pointer arithmetic to calculate the product of every element in the array.

(14 Marks)

2: a) Describe with the aid of a diagram what is meant by a linked list.

(5 marks)

b) How in C do linked lists achieve greater economies of space in comparison to arrays.

(6 marks)

c) Describe, with the aid of diagrams what happens when you
i) insert an element into a linked list and
ii) delete an element from a list.

In both cases say what happens to the space the node occupies.

(10 marks)

d) Describe in your own words with the aid of a diagram what the following piece of code does.
Your answer should make reference to the way parameters are passed to the function.
Assume the node is defined as follows:

```
struct node {
int data;
struct node* next;
};
```

```
// Function

       int Length(struct node* head) {
       int count = 0;
       struct node* current = head;
       while (current != NULL) {
       count++;
       current = current->next;
       }
       return(count);
       }
```

(14 marks)

3:

a) Outline two situations when synchronization between processes is necessary.

(6 Marks)

b) What is a Critical Section? Describe two potential consequences of two competing processes entering a critical section simultaneously.

(5 Marks)

c) Explain how semaphores may be used to prevent simultaneous access to a critical section.

(5 Marks)

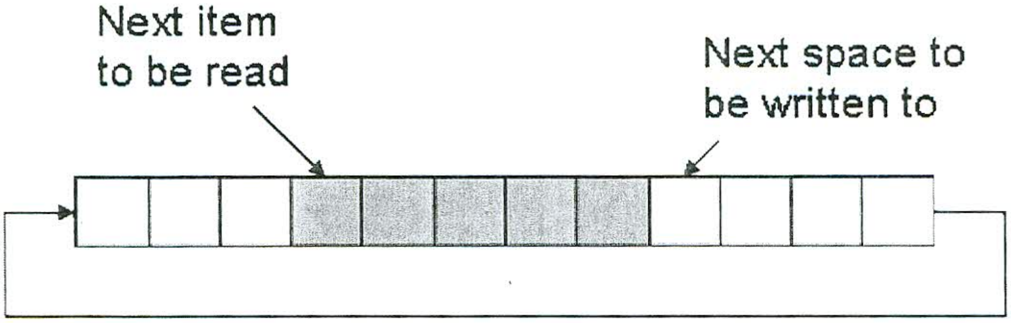
d) Describe two differences between a “test and set” instruction and a semaphore operation.

(5 Marks)

e) The following problem involving bounded buffers is a classic ***consumer-producer*** problem. Bounded Buffers are used to even out speed differences between producers and consumers

- A ***bounded buffer*** can hold up to N items

- Producer threads produce values to be placed in the buffer, consumer threads consume them from the buffer.



- Up to N producers can add N items to an empty buffer before blocking.
- Producers cannot add to a full Buffer.
- Up to N consumers can remove N items from a full buffer before blocking.
- Consumers cannot take from a n empty buffer
- Producers and consumers cannot access the buffer at the same time.
-

Describe how the following algorithms, for both the consumer and producer processes work. and satisfy the constraints of the bounded buffer problem.

- Initially itemsAvailable = 0, spaceAvailable = N

```
insert(Item i) {                                Item remove() {
   wait(spaceAvailable);                   wait(itemsAvailable);
   wait(mutex);                            wait(mutex);
   doInsert(i);                         Item i = doRemove();
   signal(mutex);                       signal(mutex);
   signal(itemsAvailable);              signal(spaceAvailable);
}                                          return i;
                                        }
```

- Note: itemsAvailable + spaceAvailable = N after each call to *insert* or *remove*

(14 Marks)

4:

a) What happens in an Operating System when a call is made

i) to open a file and

ii) to close a file?

(6 Marks)

b) What kind of information can be retrieved using the stat function?

(7 Marks)

c) What is the purpose of the following UNIX system struct and what are each of its fields for?

```
struct dirent
{   u-long d_ino;
     u_short d_reclen;
 u_short d_namelen ;
char  d_name[MAXNAMLEN+1] ;
}
```

(8 Marks)

d) Describe what the following piece of code does.

```
#include <string.h>
#include <sys/types.h>
#include <sys/dir.h>
int search (char* file, char* dir){
DIR  *dirptr=opendir(dir);
struct dirent *entry = readdir(dirptr);
while (entry != NULL)
       { if ( strlen(entry->d_name) == strlen(file) &&
(strcmp(entry->d_name, file) == 0)
          return 0; /* return success */
          entry = readdir(dirptr);     }
   return 1;
 /* return failure}
```

(14 Marks)