W228/304C

DUBLIN INSTITUTE OF TECHNOLOGY
KEVIN STREET, DUBLIN 8.

BSc. (Hons.) Degree in Computer Science

YEAR 3

SEMESTER 1 EXAMINATIONS 2014/15

CLIENT SERVER PROGRAMMING [CMPU3006]

Mr. D. Bourke

Dr. D. Lillis

Mr. P. Collins

Tuesday, 6th January 9:30 a.m.- 11:30 a.m.

Attempt Section A

and any four questions from Section B.

Section A carries 40 marks and

Section B carries 60 marks.

Section A

1. Refer to Figure 1 – *A snippet of code*. There are fourteen deliberate errors introduced in this code; eleven of the errors are identified with **XXXX** and three relate to the sequencing of the Socket Primitive calls.

   You are required to:

   (i) Replace the values **XXXX** with the correct code. Use the line numbers to identify where your corrections are made. Note the text "**XXXX**" is merely a placeholder and does not signify the same code to be replaced in each instance. (10 marks)

   (ii) Identify the correct line number where the out-of-sequence Socket Primitive calls (and any other lines normally associated with that primitive call) should be placed. (6 marks)

   (iii) Identify the type of application this is: client or server. Also, identify what application layer protocol it is adhering to and the next socket primitive to be called in relation to this application layer protocol. (4 marks)

```
1.  if (connect(XXXX, (SA *) &servaddr, sizeof(servaddr)) < 0)
2.    err_sys("connect error");
3.
4.  if ( (hp = XXXX("www.dit.ie")) == NULL)
5.    err_quit("hostname error", hstrerror(h_errno));
6.
7.  snprintf(sbuff, sizeof(sbuff),"HELO aisling.student.comp.dit.ie\r\n");
8.  Write(XXXX, XXXX, strlen(XXXX));
9.
10. pptr = hp->h_addr_list;
11.
12. bzero(&XXXX, sizeof(XXXX));
13. servaddr.sin_family = AF_INET;
14. servaddr.sin_port   = XXXX(atoi(argv[2]));
15. memcpy(&servaddr.sin_addr, *XXXX, sizeof(*XXXX));
16.
17. if ( (sockfd = socket(XXXX, SOCK_STREAM, 0)) < 0)
18.   err_sys("socket error");
```

Figure 1: A *snippet* of code

2. Write pseudocode for a *daytime server* application.

   In your code identify the following:
   - All variables, structures and buffers used in relation to the sockets API primitives,
   - The correct sequence of *socket* primitives used throughout. Precise arguments are not required to be identified.

   No error checking is required. (20 marks)

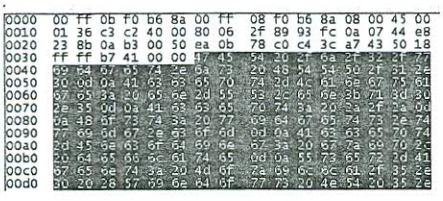
Section B

3. Refer to Figure 2 – *Wireshark extracts*. This figure shows the raw data from a captured frame as it would appear in the Wireshark application. Different portions of the frame (labelled Portion A, B, C and D) have been highlighted (the shaded areas). Encapsulation can be clearly seen in these extracts.

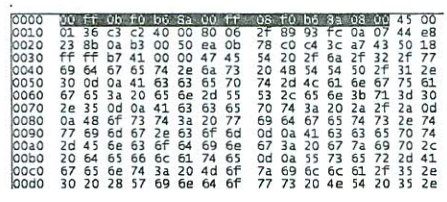
(i) Explain the concept of *encapulsation*. (3 marks)

(ii) Identify which portions (A, B, C or D) relate to: the *application layer* data, the *Frame* Header, the *IP* header and the *TCP* header. Justify your answer. (8 marks)

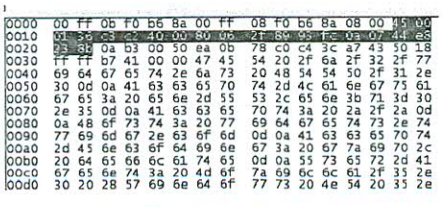
(iii) What terms are used to describe the PDUs at each layer? (4 marks)



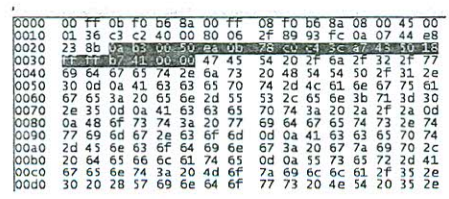
Portion A



Portion B



Portion C



Portion D

Figure 2 – *Wireshark extracts*.

4. The following snippet of code relates to the *read* socket primitive:

```
while ( (n = read(sockfd, rbuff, MAXLINE)) > 0)
{
   rbuff[n] = 0;
   if (fputs(rbuff, stdout) == EOF)
        err_sys("fputs error");
 }
```

(i) What do the first two arguments refer to? (2 marks)

(ii) In terms of the protocol layers, explain where the data is actually coming from. (3 marks)

(iii) Why is the *read* primitive placed inside a *while* loop? (5 marks)

(iv) What range values can 'n' be assigned (positive, negative etc.) and what condition does each indicate? (5 marks)

5. In relation to the use of HTTP:

   (i) What is the main purpose of HTTP? (3 marks)

   (ii) Identify two differences between HTTP/1.0 and HTTP/1.1. (2 marks)

   (iii) Explain the basic steps undertaken within the browser after a user types an address at the address bar. In your answer identify each of the components involved in the process and what its basic role is. (10 marks)

6. In relation to the *Accept* Primitive:

   (i) Explain the purpose of this primitive. (4 marks)

   (ii) Which side, client or server calls this primitive and which primitive must the other side call in order for *Accept* to return? (2 marks)

   (iii) What does this primitive return? In your answer explain what this returned entity is used for and the rational behind this approach in terms of accepting client connection requests and closing connections. (9 marks)

7. Refer to Figure 3 – *TCP State Transitions*. This figure shows TCP state transitions for applications performing *active* and *passive* connection openings.

   (i) Identify which application, client or server, performs each type of connection opening. Justify your answer. (3 marks)

   (ii) Identify which transitions, dashed or solid lines relate to each type of connection opening. (3 marks)

   (iii) Reproduce the diagram in your answer book. Identify on the diagram the TCP messages and/or acknowledgments that must be sent/received in order to move from one state to another. (9 marks)

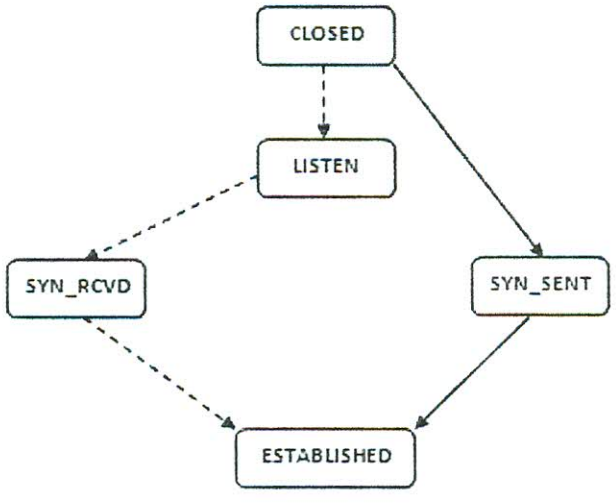


Figure 3 – *TCP State Transitions*.