Instructions

Answer as many questions as you can in the time allowed. There are 106 marks on this paper. A reasonable target is 90 marks.

Write answers to Section A on the question paper and hand it in with your answers to Section B.

Name

Student Number
Section A

Short Answer Questions

Write your answers on this sheet in the space provided. Except where the question clearly indicates the number of answers expected, the answers to each part in Section A should require no more than two concise sentences. In some cases, a single word or expression will suffice. If you need more space or wish to change your answer, use your answer book. Make sure to number the question clearly in your answer book.

Program design

1. Give two reasons why the choice of names in a program is important.  

2. Do you think it was a good idea to insist on ANSI standard C in this course? Give two reasons for your answer.

3. Name the two primary components of a software class.

4. Why do the elements of the C struct that contains the attributes of a class appear in the implementation only?
5 What should happen to the pre-conditions for a function in the implementation?

6 How many constructors may a class have?

C language
7 The C for statement has the form:

```c
for(expression1; expression2; expression3) statement;
```

Explain the meaning of this for statement by writing out a set of statements using a `while` statement, which have the same effect in a program.

8 In C, what is the value of the index of the first value of any array?

9 If `a` and `b` are both `double` variables, why is this statement - at best - unwise and generally downright dangerous?

```c
if ( a == b ) printf("They’re equal\n");
```

10 Suggest a much better way of writing the statement in the previous question.
11. Why was the ANSI C library function
   \[ \text{char *gets(char *s);} \]
   banned from this course as too dangerous for any professional software
   engineer to use?

12. What does the following statement do:
   \[ x = y = 0; \]

13. Give a formal explanation for your answer to the previous question.

14. Why was it suggested that learning C’s operator precedence rules is a waste
   of time? There are two parts to the full answer here!

15. On a typical 32-bit modern processor, what range of values can variables x and y take?
   \[ \text{int x; unsigned int y; } \]
   You may use any convenient notation, e.g., powers of 10, powers of 2, hexadecimal, etc,
   as long as it’s clear which notation you are using.

   \[ \begin{array}{ll}
   \text{Minimum} & \text{Maximum} \\
   x & \text{ } \\
   y & \text{ } \\
   \end{array} \]
16. Why does the C compiler accept the statement
   \[ \text{if}(x = y) \{ \text{some statements}; \} \]
when the programmer intended to write:
   \[ \text{if}(x == y) \{ \text{some statements}; \} \]
instead of producing an error?

17. C programmers often use the following device to write a loop that executes \( n \) times:
   \[ \text{while}(n--) \{ \text{some statements}; \} \]
Explain why this loop terminates after the \( n^{th} \) iteration.

18. If I write variables of an enumerated type into a file:
   \[
   \text{typedef enum (black, blue, red, yellow) Colour;}
   \text{Colour a[N];}
   \ldots
   \text{write(f, a, N*sizeof(Colour));}
   \]
   and then transmit my program to another computer where it is read back:
   \[
   \text{typedef enum (black, blue, red, yellow) Colour;}
   \text{Colour a[N];}
   \ldots
   \text{read(f, a, N*sizeof(Colour));}
   \]
why should I not be surprised if my program behaves oddly?
Examine the following fragment of code:

```c
switch( x ) {
    case 1: printf("red ");
    case 2: printf("black "); break;
    case 3: printf("blue ");
    default: printf("unknown ");
}
```

What does it print for the following values of `x`?

- 0
- 1
- 2
- 3

Examine the following code fragment and write down the output produced by the `printf` statements:

```c
char *s;
s = "CLP110";
printf("s = %d\n", s);
printf("A %c\n", *s);
printf("B %c\n", *(s+2) );
printf("C %c\n", s[3] );
```

The characters of the string are stored in memory (addresses in decimal) as follows:

```
1000 1001 1002 1003 1004 1005 1006
'C' 'L' 'P' '1' '1' '0' '\0'
```

Why is the following ANSI C statement often unsafe?

```c
strcpy( dest_buf, src_buf );
```
22 Write down a safer alternative to the statement in the previous question. Why is it safer?

23 Verifying functions
Why is it important not to rely on a user’s input when testing a function (other than one which depends on user input for its operation)?

24 Why is the concept of equivalence classes useful in verifying functions?
Section B

QUESTION B1  
10 marks

A class of vehicles needs an attribute describing the body type: possible body types are sedan, coupe, hatchback, etc.

a) What type of variable would you use to identify each vehicle’s body type?

b) Show how this variable would be declared.

c) Write a function:

```c
void print_body_type( …. );
```

that will print out, in some human readable language (ie not a computer code!), the body type. You should fill in the argument(s) yourself.

QUESTION B2  
25 marks

A clerk managing the loading of a fleet of freighter aircraft needs to monitor the loading of each craft and determine whether it is fully loaded or not. Aircraft are constrained both in the total volume of cargo and in the total mass of the cargo. Each aircraft in the fleet has a different maximum cargo volume and maximum cargo mass. In addition, at the beginning of each flight, the clerk will specify the distance to the destination. From this distance, the fuel mass for the flight can be calculated. This fuel mass has to be included in the total cargo mass - but because the fuel tanks are already built into the wings, the volume of the fuel is not considered in the loading calculations.

Rules

i) Each aircraft has a known and fixed: maximum load mass at takeoff (kgs), maximum cargo volume (m3) and fuel efficiency (kg/km). These are specified to the system when the aircraft is added to the fleet and never change.

ii) At the beginning of each flight, the clerk wants to reset the aircraft’s loading and specify the distance to the destination in kms.

iii) As he loads the aircraft, the clerk will specify the mass and volume of each shipment.

iv) So that he can find out whether any aircraft will be able to take a given shipment, the clerk wants to be able to find out the remaining space and the remaining volume on an aircraft.

v) Whenever he adds a new cargo item to the load, the clerk wants an indication whether this new item has filled the aircraft or not. Naturally, if it would overload the aircraft, the item is not added to the aircraft’s load.

Write a software model for an aircraft that will enable the clerk to keep track of the current loading status of any aircraft in the fleet. This model should be designed to be incorporated in a larger program which handles the clerk’s input: since this will use the latest Japanese neuro-fuzzy-brain-wave interpretation technology, you do not need to
concern yourselves with details of this higher level program. Just provide a model that will fit well into any environment.
QUESTION B3

Your colleague has written a function:

```c
int fgrep( char *file_name, char *string );
/* Return the number of times "string"
   is found in the file "file_name"
   Pre-cond: file_name is a valid string and
   String is a valid string
   Post-cond: returns count of occurrences of
   string in the file or
   -1 if the file does not exist
   */
```

and you have been assigned the task of testing it. This function is designed to find strings in files – so it opens the file, reads all the text in it and counts the number of times it can find the string.

Work out a strategy for testing this function: decide how many test files you will need and what they should contain.

a) Make a table of the test files and summarise the essential details of their contents, eg

<table>
<thead>
<tr>
<th>File Name</th>
<th>Number of lines</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1.DAT</td>
<td>108</td>
<td>String “abcd” in lines 1, 23 and 58</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

b) Outline the code for the key part of a program that will test the function. This outline should include any data structures to be used and the essential code for running the tests.

Note: To carry out this exercise thoroughly, you will actually need to perform a very large number of tests. So you will not be examined on completeness of your tests but rather on a systematic approach to testing the function. It is suggested that you allocate yourself a fixed amount of time to answer this question and list about 10 test cases. You can earn full marks for sets of tests (containing no more than 10 tests in total) which demonstrate a systematic strategy which, if carried to completion, would rigorously test the function.

**DO NOT WRITE CODE FOR fgrep - ASSUME THAT HAS BEEN DONE FOR YOU!**