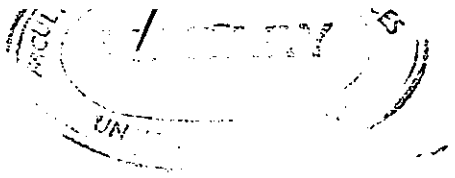


UNIVERSITY OF GHANA

BACHELOR OF SCIENCE IN ENGINEERING

SECOND SEMESTER EXAMINATIONS, 2010/2011

FAEN 112 C PROGRAMMING (2 Credits)

INSTRUCTIONS:

ANSWER ALL QUESTIONS IN THE ANSWER BOOKLET

EACH QUESTION CARRIES 20 MARKS

TIME ALLOWED: TWO (2) HOURS

Q1. (A) Fill in the blanks in each of the following sentences: [10 marks]

a) The ———————— loop always iterates at least once.

b) A loop that does not have a way of stopping is a(n) ———————— loop.

c) Each repetition of a loop is known as a(n) ———————— .

d) The header file ———————— contains the prototypes for printf () and scanf() functions.

e) Every complete C statement ends with a ———————— .

f) Every C program must have a function ———————— .

g) Words that have a special meaning in a programming language are called ———————— .

h) Words or names defined by the programmer are called ———————— .

i)———————— are on the same level of precedence as the division operator.

j)The header file———————— .contains the prototypes for the sqrt() and log10( ) functions.

EXAMINER: P. OKAE

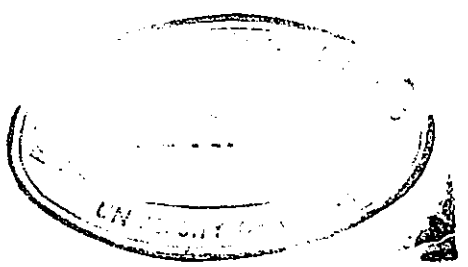
(B) [10 marks]

In each of the following, match the character (s) on the **left** with the *correct description* on the **right**.

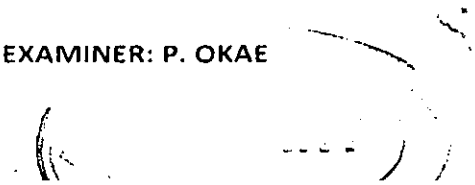
Character	Description
/* */	a) Encloses a string of characters, such as a message that is to be printed on the screen.
#	b) Encloses a group of statements, such as the contents of a function.
< >	c) Marks the beginning of a preprocessor directive.
( )	d) Marks the end of a complete programming statement.
{ }	e) Used in naming a function.
" "	f) Encloses a filename when used with the *#* include directive.
;	g) Marks the beginning and end of a comment.
%d	h) Specifies a *double* value.
%f	i) Specifies a *string* value.
%s	j) Specifies an *integer* value.

Q2. a) What will the following program display?

```
# include <stdio.h>
int main()
{
   int funny = 7, serious = 15;
  funny = serious % 2:
 if ( funny ! = 1)
 {
   funny = 0;
   serious = 0;
```



EXAMINER: P. OKAE



```
  }
else if ( funny = = 2 )
{
  funny = 10;
  serious = 10;
}
else
 {
  funny = 1;
  serious = 1;
 }
  printf("%d  \t  %d\n", funny, serious);
  return 0;
}
```

[5 marks]

b) Write an *if statement* that prints the following messages:

(i) "The number is valid" if the variable *grade* is greater or equal to 90.

(ii) "The number is valid" if the variable *temperature* is within the range -50 through 150.

(iii) "The number is invalid" if the variable *hours* is less than 80. [3 marks]

c) (i) Write an *if statement* that assigns 100 to x when y is equal to 0.

(ii) Write an *if else statement* that assigns 0 to x when y is equal to 10. Otherwise it should assign 1 to x. [2 marks]

d) Write down the equivalents (expanded forms) of the following compound assignment statements:

(i) total += 100; [1 mark]

(ii) value - = 20; [1 mark]

(iii) perDiem % = 5; [1 mark]

(iv) answer / = 1000; [1 mark]

(v) mooresLaw * = 2; [1 mark]

c) Assuming the following are expressions in a C program, calculate their values according to the precedence rules of the C language.

(i) 5 + 2 * 4 [1 mark]

(ii) 10 / 2 – 3 [1 mark]

(iii) 8 + 12 * 2 – 4 [1 mark]

(iv) 4 + 17 % 2 – 1 [1 mark]

(v) 6 – 3 * 2 + 7 – 1 [1 mark]

Q3. In a college freshman computer programming class, students were asked to write a simple program to mimic the *perimeter* of their computer laboratory floor which is *rectangular* in shape.

a) Design the *algorithm* for the task above. [5 marks]

b) Convert the algorithm above into a simple C program that prints to the screen the perimeter of the computer laboratory floor. [5 marks]

c) State which values of the control variable x are printed by each of the following *for statements*:

i. for (x = 2; x < = 13; x + = 2)

```
printf("%d\n", x );
```
[1 mark]

ii. for (x = 5; x < = 22; x + = 7)

```
printf( "%d\n", x);
```
[1 mark]

iii. for (x = 3; x < = 15; x + = 3)

```
printf( "%d\n", x);
```
[1 mark]

iv. for (x = 1; x < = 5; x + = 7)

```
printf( "%d\n", x );
```
[1 mark]

v. for (x = 12; x > = 2; x - = 3)

```
printf( "%d\n", x )
```
[1 mark]

d) int x = 1, y = 1;

```
while ( x <= 4)
{
   printf ( "%d plus %d equals %d\n", x, y, (x + y) );
   x++;
}
```

In the code given in Q3(d) above:

i. identify the loop control variable (LCV),

ii. write down the output as it appears on the computer screen. [5 marks]

Q4. a) Draw a simple *block diagram* that shows the processes (appropriate order) by which a source code is translated into executable code. [7 marks]

b) Define (or explain) the following types of errors as applied to computing:

(i) syntax error. [2 marks]

(ii) run-time error. [2 marks]

(iii) logical error. [2 marks]

c) Write down the output of the following program as it will appear on the computer screen.

```
/* This program has three functions: main, first and second*/
   # include <stdio.h>
    int main()
    {
     printf("I am starting in function main.\n");
     first();                                  /* call function first*/
     second();                                 /* call function second*/
     first();                                  /*call function first*/
     printf("Back in function main again.\n");
    return 0;
    }
   void first()
```

```
{
  printf ("I am now inside the function first.\n");
}
void second()
{
  printf(" I am now inside the function second.\n");
}
```

[7 marks]

Q5. a) State the header files that contain the prototypes of each of the following functions:
(i) printf() and scanf(),

(ii) sqrt() and log10(),

(iii) rand() [5 marks]

b) Assume the following variables are defined:

```
int age:

double pay:

char section;
```

Write a single scanf() statement that will read input into each of these variables above. [3 marks]

c) (i) Define the *float* variables *temp*, *weight*, and *age* in one statement? [2 marks]

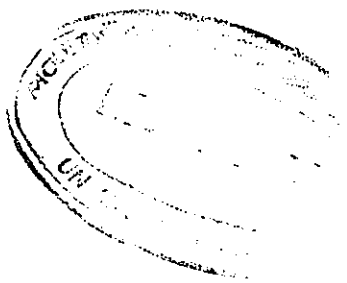
(ii) Define the same statement in (i) above in *three* separate statements? [2 marks]

d) In the following program, write in your answer booklet the correct order in which the statements are executed, i.e. use the numbers 1, 2, 3 etc., to indicate the logical order of execution. Thus if, for example, statement (ii) is executed first, write (ii) ⟶ 1

```
#include <stdio.h>

int main()

{

    int num1, num2;

    (i) ----    printf ("Please enter two integers.\n");

    (ii) ----   scanf ( "%d%d", &num1, &num2);

    (iii)----   func (num1, num2);

    (iv)----   printf ("The two integers are %d, %d\n", num1, num2);

    (v) ----   return 0;                    /* indicates successful termination*/
```

```
}
    void func (int val1, int val2)          /* function definition*/
    {
       int val3, val4;
(vi)    ----    val3 = val1 + val2;
(vii)   ----    val4 = val1 * val2;
(viii)  ----    printf ("The sum and product are %d and %d\n", val3, val4);
    }
```

[8 marks]



EXAMINER: P. OKAE