Université d'Ottawa
Faculté de génie

École d'ingénierie et de
technologie de l'information

uOttawa

L'Université canadienne
Canada's university

University of Ottawa
Faculty of Engineering

School of Information
Technology and Engineering

GNG 1106 **Examen de mi-session / Midterm Examination** **March 5th, 2011**

Time allowed: 90 minutes
Closed book examination
Non-programmable calculators are allowed

Attempt all questions
Questions carry the weights indicated
The total number of points for the examination is 100

Answer the questions in the spaces provided
Use both sides of these sheets if necessary

Les réponses en français sont acceptées.

Name: _____ **Student Number:** _____

Part 1:	30
Part 2:	35
Part 3:	35
Total:	100

Do not write in this box!

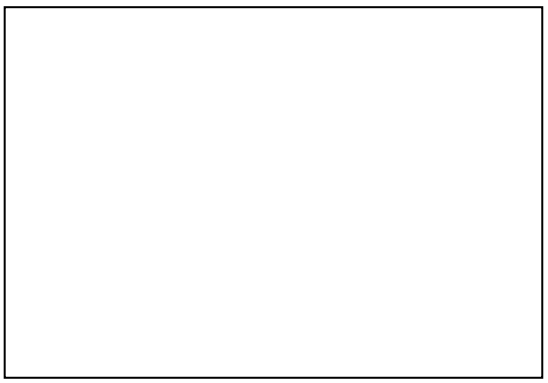
Part 1 – Tracing *(30 points)*

A) (4 points)

```
int x = 130;

if(x >= 0 && x <= 100)
   printf("Valid grade");
else
{
if (x < 0)
     printf("Under range!!");
else
     printf("Over range!!");
}
```

Output Screen

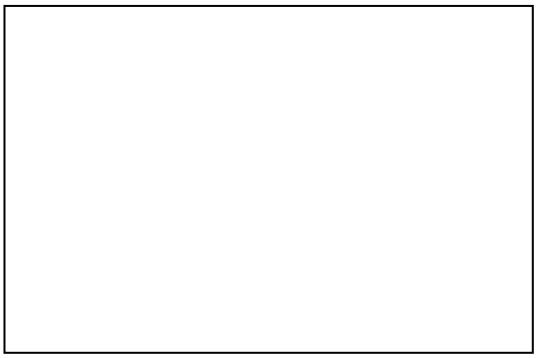


B) (6 points)

```
i = 1;

while (i < 10)
{
   if (i%3 == 0)
     printf("%d \n", i);
   else
     printf("%d\t", i);
   i++;
}
```

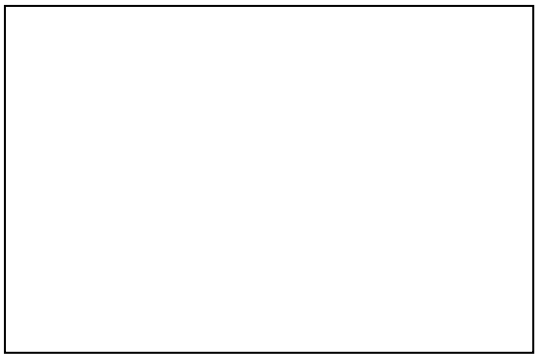
Output Screen



C) (8 points)

```
for (i = 4; i > 0; i--)
{
   for (j = 0; j < i; j++)
        printf("*");
   printf("%d\n", i+j);
}
```

Output Screen



D) (12 points)

Assume the following declaration for the array g:

```
int i, j,  g[3][3] = {{0,0,0},{1,1,1},{2,2,2}};
```

Write down the screen output generated by each of the following C codes:

```
sum = 0;
for (i = 0; i <= 2; i++)
  {
     for (j = 0; j <= 2; j++)
          sum = sum + g[i][j];
  }
printf ("The value is: %d", sum);
```

Output Screen

```
sum = 1;
for (i = 1; i <= 2; i++)
  {
     for (j = 0; j <= 1; j++)
          sum = sum * g[i][j];
  }
printf ("The value is: %d", sum);
```

Output Screen

```
sum = 0;
for ( i = 0; i <= 2; i++)
   sum = sum + g[i][1];

printf ("The value is: %d", sum);
```

Output Screen

Part 2 - Debugging *(35 points)*

A) (17 points)

This program is supposed to compute the product of the entries for a one dimensional array. Find six errors (syntax and/or Logical) in the program. Find the program line numbers where these errors occur and explain them.

```
1.      #include <stdio.h>
2.      #define N 6
3.
4.      void Function (float array[]);
5.
6.      int mian()
7.      {
8.        int i;
9.        float array[N];
10.       float Product;
11.
12.       for (i=0; i<=N; i++)
13.             scanf("%f", &array(i));
14.
15.       Product = Function(array);
16.
17.       printf("%f\n", Product);
18.
19.       Return (0);
20.     }
21.
22.     void Function (float tab[])
23.     {
24.       float  product=0;
25.
26.       for (i=0; i<N; i++)
27.          product= product*tab[i];
28.
29.       return(product);
30.     }
```

a) Line number:

_____

_____

b) Line number:

_____

_____

c) Line number:

d) Line number:

e) Line number:

f) Line number:

B) (9 points)

This code is supposed to exit when the user inputs 0 or repeat adding two numbers if the user inputs 1. Find three logical and/or syntax errors in this program.

```
1.    int choice, num1, num2:
2.
3.    do
4.     {
5.        printf("Enter a number: ");
6.        scanf(“%d”, &num1);
7.        printf("Enter another number: ");
8.        scanf(“%d”, &num2);
9.        printf("Their sum is %d\n" , (num1+num2));
10.       printf("Enter 1 to repeat, 0 to exit");
11.       scanf(“%d”, &choice)
12.    } while (choice == 0)
```

a) Line number:

b) Line number:

c) Line number:

C) (9 points)
There are three errors in this program.

```
1.    int a = - 75;
2.
3.    if (a ≥ 0)
4.         if (a > 1000)
5.              a = 0;
6.         else
7.           if (a < 500);
8.              a = a + 2;
9.         else
10.             a = a*10;
      else
11.     a = a + 3;
12.     printf('The output is %f', a);
```

a) Numéro de ligne:/Line number:

b) Numéro de ligne:/Line number:

c) Ligne numéro:/Line number:

Part 3 – Problem Solving: Computing Bank Investments (35 points)

a) 2 points; b) 3 points; c) 6 points; d) 12 points; e) 12 points

Your project is to design a software tool to compute what amount A will be generated when an amount P is invested (the principal) and an annual interest rate, i, for all the years from 1 to n. The software provides two options to the user, a first that simply outputs the amount A along with the values P, i, and n provided by the user and the second where a table is generated that shows how the amount A progresses each year (from year 1 to year n) also with the values P, i, and n provided by the user.

The following formula relates A, P, i, and n: $A = P(1+i)^n$

Here is a description of the software subprograms to be used in the software:

- `displayMenu()`: a subprogram that displays the main menu to the user and return only a valid choice to the main subprogram (valid options are 1, 2, and -1).
- `computeAmount(A, P, i, n)`: a subprogram that computes all the amounts at the years 1 to n and stores them in the array `A`. The subprogram does not return any value as the amount values will be stored in the array.
- `prinTable(A, P, i, n)`: a subprogram that prints the values `P`, `i` and `n`. Then it prints a table of the years 1 to n and the corresponding amount earned (from the array `A`). The sub-program should be void!
- `main`(): the main subprogram first calls the `displayMenu()` subprogram and according to the returned value calls either the `computeAmount(P, i, n)` or the `prinTable (P, i, n)` subprograms. The main keeps looping back until the user inputs -1 to exit!

a) **Step 1** – *Statement of the problem*: provide a clear description of the problem to be solved.

Answer (2 points):

b) **Step 2b** – *Input and Output Description*: Give a diagram that shows all inputs and outputs of the software tool.

Answer (3 points):

c) **Step 3a** – *Test Cases*: Give three test cases for the software tool covering the followings cases: (1) when the user inputs invalid option, (2) when the user picks option 1 (computing and printing A), and (3) when the user picks option 2 (printing a table of amounts (A) over the years).

Answer (6 points):

d) **Step 3b** – *Algorithm design*:

Write the pseudocode for the `displayMenu()` and the `computeAmount(A, P, i, n)` subprograms (C code is not accepted for this part!):

Answer (5 points + 7 points):

displayMenu( )

computeAmount(A, P, i, n)

e) Step 4 – *Implementation*:

Translate the following pseudocode to C code.

main ( ) (8 points)

```
Declare A, p, n, i and choice

Repeat
        Print "Enter values for p, n, and i"
        Read values into p, n, and i
        computeAmount(A, P, i, n)
        Assign displayMenu( ) to choice
        If choice is equal to 1
                Print "P = ", p, "i = ", i, "n = ", n
                Print "The final amount A = ", A[n-1]
        Else if choice is equal to 2
                prinTable(A, P, i, n)
        else
                print "program terminating …"
    while choice is not equal to -1
```

C Code

```
int main()
{









return (0);
}
```

printTable (A, p, i, n) (4 points)

Declare ctr
Assign 0 to ctr
Print "P = ", p, "i = ", i, "n = ", n
Print "Year" tab "Amount"
Repeat while ctr is less than n
print ctr+1, A[ctr], newline
Increment ctr

C Code

```
void printTable(float A[], float P, int i, int n)
{









}
```

Extra page (you can detach this page and use it as scratch)