GNG1101/1501 **Examen Final Examination - Solved** **1999**

Time allowed: 3 hours.
Closed book examination.
NO Calculator 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.
Additional answer books are available if required.

Name/ Nom: **Pierre Berini** **Student number/ Nombre d'étudiant(e):** _____

Question 1	**25/25**
Question 2	**25/25**
Question 3	**25/25**
Question 4	**25/25**
Total	**100/100**

Question 1a) ***(7 points)***
Write down what is printed out in the following program.

Donnez la sortie générée par le programme suivant.

```
#include <stdio.h>
void main()
{
      int i, n, s;

      for(n=50; n<60; n++)
      {
            s=1;

            for(i=2; i<8 && s>0; i++)
            {
                  if(n%i==0) s=0;
                  if(i%2==1) i++;
            }
            if(s==1) printf("%d\n", n);
      }
}
```

Answer to Question 1a), Réponse à la question 1a):

53

59

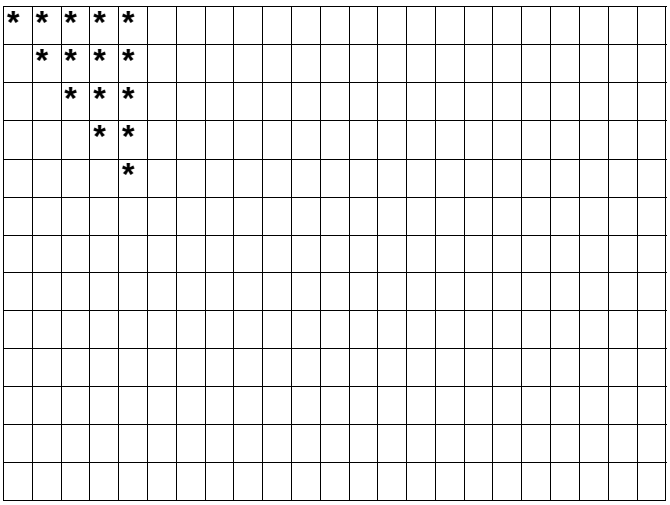
Question 1b) ***(8 points)***
Write down what is printed out in the following program. Use the grid on the right. Each square represents one character position on the screen.

Donnez la sortie générée par le programme suivant.

```
#include <stdio.h>
Void main()
{
   int i, j;

   i=1;
   while(i<6)
   {
      for(j=1; j<i; j++)
         printf(" ");
      for(j=i++; j<6; j++)
         printf("*");
      printf("\n");
   }
}
```

Answer to Question / Réponse à la question 1b):



Question 1c) ***(10 points)***

Write down what is printed out in the following program, taking the first column below as the keyboard entries. Indicate what, if anything is printed out after each entry.
Donnez la sortie générée par le programme suivant.

```
#include <stdio.h>
void main()
{
   int i, mult=0, decade=1;

   do
   {
      scanf("%d", &i);
      switch(i)
      {
         case 0:
         case 1:
         case 2:
         case 3:
            mult=mult+decade*i;
            decade=decade*10;
            break;
         case 4:
            printf("4: %d\n", mult);
            mult=0; decade=1;
            break;
         case 5:
            printf("5: %d\n", mult);
            mult=0; decade=1;
            break;
         case 6:
            scanf("%d", &i);
            if(i!=4)
            {
               printf("ERR\n");
            }
            else
            {
               printf("6: %d\n", mult);
               mult=0; decade=1;
            }
            break;
         default:
            printf("ERR\n");
      }
   }while(i>0);
}
```

Answer to Question 1c), Réponse à la question 1c):

3	
4	**4:3**
1	
2	
5	**5: 21**
2	
6	
4	**6: 2**
9	**ERR**
0	

Question 2a) ***(10 points)***

Find and explain the error in each of the following program segments.

Assume:

```
int *zPtr;
int *sPtr=NULL;
int number, i;
int z[5] = (1, 2, 3, 4, 5};
sPtr = z;
```

a)
```
zPtr++;
```

zPtr has not been initialized.

b)
```
number = sPtr;
```

Attempts to store an address into a non-pointer variable.

c)
```
number = *sPtr[2];
```

Attempts to perform dereferencing twice.

d)
```
for (i = 0; i <= 5; i++)
      printf("%d ", sPtr[i]);
```

The case i=5 goes beyond the array limits.

e)
```
++z;
```

z is the name of an array so this pointer cannot be modified.

2(b) ***(5 points)***

The statement `int x, *ptr=&x;` places the variable `x` into memory at address 8688 and the other variable at 8684. Write down the output you expect for the following code:

```
#include <stdio.h>
void func(int);
void main()
{
 int x, *ptr=&x;

 x=132;
 func(*ptr);
 printf("%d\n", x);
}
void func(int y)
{
 y=y+2;
 printf("%d\n", y);
}
```

Answers to Question 2 b):

134
132

```
#include <stdio.h>
void func(int*);
void main()
{
 int x, *ptr=&x;

 x=132;
 func(ptr);
 printf("%d\n", x);
}
void func(int *y)
{
 y=y+2;
 printf("%d\n", y);
}
```

8692
132

2(c) ***(10 points)***

The following code operates on an array using the function `func`. Write down the output you expect.

```
#include <stdio.h>
#define M 2
#define N 3
void func(int [][N], int*, int);
void main()
{
 int a[M][N]={{1, 4, 6},{3, 2}};
 int i, j;

 func(a, &a[0][0], 2);
 for(i=0; i<M; i++)
 {
  for(j=0; j<N; j++)
      printf("%d ", a[i][j]);
  printf("\n");
 }
}
void func(int x[][N], int *y, int z)
{
 int i;
 for(i=0; i<M; i++)
      x[i][z]=*(y+i);
}
```

Answers to Question 2 c):

1 4 1
3 2 4

Question 3 ***(25 points)***

3(a) ***(15 points)***

Write down the **exact** output you expect for the following code. Use the grid, each square of the grid represents one character position on the screen.

```
#include <stdio.h>
#include <math.h>
#define CONST 2000
#define PI 3.14159
void main()
{
      int i;
      float a=3.9, b=3.33333, c;
      double d=1.0e-200;
      long z=2000000;

/* a) */ i=a;                                          printf("a) %+6d\n", i);
/* b) */                                              printf("b) %-6d\n", i);
/* c) */                                              printf("b) %9.1e\n", d);
/* d) */ a=(float)z/CONST;                            printf("d) %f\n", a);
/* e) */ i=a+b;                                       printf("e) %d\n", i);
/* f) */ a=a/b;                                       printf("f) %.1f\n", a);
/* g) */ a=sin(PI);                                   printf("g) %5.1f\n", a);
/* h) */ i=5; d=++i*2.0;                              printf("h) %f\n", d);
/* i) */ i=acos(cos(PI));                             printf("i) %d\n", i);
/* j) */ i=3; a=(float)i;                             printf("j) %f\n", a);
/* k) */ b=4.0; c=sqrt(pow(a,2.0)+pow(b,2.0));        printf("k) %f\n", c);
/* l) */ c=floor(PI);                                 printf("l) %.1f\n", c);
/* m) */ a=10+6/3+1;                                  printf("m) %.3f\n", a);
/* n) */ a=(10*6)/(3+1);                              printf("m) %.2f\n", a);
/* o) */ i=200/5/4+1;                                 printf("o) %d\n", i);
}
```

Answer to Question 3a), Réponse à la question 3a):

a	)						+	3																		
b	)		3																							
b	)			1	.	0	e	-	2	0	0															
d	)		1	0	0	0	.	0	0	0	0	0	0													
e	)		1	0	0	3																				
f	)		3	0	0	.	0																			
g	)				0	.	0																			
h	)		1	2	.	0	0	0	0	0	0															
i	)		3																							
j	)		3	.	0	0	0	0	0	0																
k	)		5	.	0	0	0	0	0	0																
l	)		3	.	0																					
m	)		1	3	.	0	0	0																		
m	)		1	5	.	0	0																			
o	)		1	1																						

3(b) ***(10 points)***

Write down the **exact** output you expect for the following code. Use the grid, each square of the grid represents one character position on the screen.

```
#include <stdio.h>
#define M 5
```

```
void main()
{
   int i, j, a[M][M], temp;

   for(i=0; i<M; i++)
      for(j=0; j<M; j++)
         a[i][j]=i;

   for(i=0; i<M; i++)
   {
      for(j=0; j<M; j++)
         printf("%d ",a[i][j]);
      printf("\n");
   }
   printf("\n");

   for(i=0; i<M; i++)
      for(j=i; j<M; j++)
      {
         temp=a[i][j];
         a[i][j]=a[j][i];
         a[j][i]=temp;
      }

   for(i=0; i<M; i++)
   {
      for(j=0; j<M; j++)
         printf("%d ",a[i][j]);
      printf("\n");
   }
}
```

Answer to Question 3b), Réponse à la question 3b):

0		**0**		**0**		**0**		**0**																	
1		**1**		**1**		**1**		**1**																	
2		**2**		**2**		**2**		**2**																	
3		**3**		**3**		**3**		**3**																	
4		**4**		**4**		**4**		**4**																	
0		**1**		**2**		**3**		**4**																	
0		**1**		**2**		**3**		**4**																	
0		**1**		**2**		**3**		**4**																	
0		**1**		**2**		**3**		**4**																	
0		**1**		**2**		**3**		**4**																	

<u>Question 4</u> ***(25 points)***

The marks scheme of the University of Ottawa is defined as in Table 1. The final percentages in a course are translated into a letter grade, each letter grade has an associated grade point value. A Grade Point Average (GPA) is calculated as a weighted average of the grades achieved in the courses of a program of studies.

%	Grade	Points
90-100	A+	10
85-89	A	9
80-84	A-	8
75-79	B+	7
70-74	B	6
66-69	C+	5
60-65	C	4
55-59	D+	3
50-54	D	2
40-49	E	1
0-39	F	0

Table 1

results.dat
C
A
A+
A+
E
.
.
.

Table 2

The results of a particular course are stored in the file **results.dat**. The file structure is as shown in Table 2. The file contains one letter grade per line. There is no indication in the file of how many grades are in it.

Write a program that reads the file **results.dat** with the structure presented. Your program must contain a single dimensional array **hist[11]**. Associate **hist[0]** with the grade F, **hist[1]** with the grade E etc. As the data is read, you are to count in **hist** the number of each grade. When all the grades have been read, the program is to print out the number of each grade read, the maximum and minimum grades of the class, and the grade point average for the class.

Follow the standard approach to problem solving.

Answer to Question 4:

Problem statement *(2 points)*

Parse and process a file containing the grades for a class.

Description inputs and outputs *(3 points)*

Inputs: The data file containing the grades for a class.
Outputs: Number of occurrences of each grade, maximum grade, minimum grade and class gpa.

Algorithm development *(7 points)*

Open the file to be processed.
If file can be opened:

Loop until the end of the file is reached:
Read in a grade from the file.
Accumulate in an array of `int`'s (`hist`) the number of occurrences of a grade.

Loop over all grades using the array hist:
If the current grade is greater than the max grade encountered to date, then memorize.
If the current grade is less than the min grade encountered to date, then memorize.
Accumulate the points.
Accumulate the total number of grades.

Compute the class gpa.
Print out to the screen the number of occurrences of each grade.
Print out to the screen the maximum grade.
Print out to the screen the minimum grade.
Print out to the screen the class gpa.

Program in C *(13 points)*

```
/*****************************************/
/* Final 99, Question 4.                 */
/* Pierre Berini.                        */
/*****************************************/
#include <stdio.h>

   void main()
{
   int ctr, maxGradePos=0, minGradePos=10,
       classSize=0, hist[11]={0};
   float gpaAve=0.0;
   char input[3];
   FILE *filePtr;

      if ( (filePtr=fopen("fin99q4.dat", "r")) == NULL)
      printf("Error opening file.\n");
      else
      {

/* Read the file and parse the grades. */
         while( (fgets(input, 3, filePtr)) != NULL)
         {
            if(input[0] == 'A')
              {if(input[1] == '+')
               hist[10]++;
               else if (input[1] == '-')
               hist[8]++;
               else
               hist[9]++;}

            else if(input[0] == 'B')
                   {if(input[1] == '+')
                    hist[7]++;
                    else
                    hist[6]++;}

            else if(input[0] == 'C')
                   {if(input[1] == '+')
                    hist[5]++;
                    else
                    hist[4]++;}

            else if(input[0] == 'D')
                   {if(input[1] == '+')
                    hist[3]++;
                    else
                    hist[2]++;}

           else if(input[0] == 'E')
                hist[1]++;

           else if(input[0] == 'F')
                hist[0]++;
         }

/* Find the maximum and minimum grades of the class */
/* and the grade point average of the class.        */
         for(ctr=0; ctr<=10; ctr++)
```

```
         {
            if(hist[ctr]>0)
            maxGradePos=ctr;

            if(hist[10-ctr]>0)
            minGradePos=10-ctr;

            gpaAve=gpaAve+ctr*hist[ctr];
            classSize=classSize+hist[ctr];
         }
      gpaAve=gpaAve/classSize;

/* Print out the number of occurences of each grade. */
      printf("Number of occurences of each grade:\n");
      printf("Number of A+: %d\n", hist[10]);
      printf("Number of A : %d\n", hist[9]);
      printf("Number of A-: %d\n", hist[8]);
      printf("Number of B+: %d\n", hist[7]);
      printf("Number of B : %d\n", hist[6]);
      printf("Number of C+: %d\n", hist[5]);
      printf("Number of C : %d\n", hist[4]);
      printf("Number of D+: %d\n", hist[3]);
      printf("Number of D : %d\n", hist[2]);
      printf("Number of E : %d\n", hist[1]);
      printf("Number of F : %d\n", hist[0]);

/* Print out the maximum grade of the class. */
         if(maxGradePos == 10) printf("Maximum class grade: A+\n");
         if(maxGradePos == 9)  printf("Maximum class grade: A\n");
         if(maxGradePos == 8)  printf("Maximum class grade: A-\n");
         if(maxGradePos == 7)  printf("Maximum class grade: B+\n");
         if(maxGradePos == 6)  printf("Maximum class grade: B\n");
         if(maxGradePos == 5)  printf("Maximum class grade: C+\n");
         if(maxGradePos == 4)  printf("Maximum class grade: C\n");
         if(maxGradePos == 3)  printf("Maximum class grade: D+\n");
         if(maxGradePos == 2)  printf("Maximum class grade: D\n");
         if(maxGradePos == 1)  printf("Maximum class grade: E\n");
         if(maxGradePos == 0)  printf("Maximum class grade: F\n");

/* Print out the minimum grade of the class. */
         if(minGradePos == 10) printf("Minimum class grade: A+\n");
         if(minGradePos == 9)  printf("Minimum class grade: A\n");
         if(minGradePos == 8)  printf("Minimum class grade: A-\n");
         if(minGradePos == 7)  printf("Minimum class grade: B+\n");
         if(minGradePos == 6)  printf("Minimum class grade: B\n");
         if(minGradePos == 5)  printf("Minimum class grade: C+\n");
         if(minGradePos == 4)  printf("Minimum class grade: C\n");
         if(minGradePos == 3)  printf("Minimum class grade: D+\n");
         if(minGradePos == 2)  printf("Minimum class grade: D\n");
         if(minGradePos == 1)  printf("Minimum class grade: E\n");
         if(minGradePos == 0)  printf("Minimum class grade: F\n");

/* Print out class GPA. */
      printf("Class GPA: %f\n", gpaAve);
   }
}
```