

CMPE 108

SAMPLE FINAL EXAM QUESTIONS

PRECEDENCE AND ASSOCIATIVITY

<u>OPERATORS</u>	<u>ASSOCIATIVITY</u>
() []	Left to right
! ++ -- + - (type)	Right to left (Unary)
* / %	Left to right
+ -	Left to right
< <= > >=	Left to right
== !=	Left to right
&&	Left to right
	Left to right
= += -= *= /= %=	Right to left
,	Left to right

QUESTIONS WITH SOLUTIONS

Q1. (10 pts) Write a C arithmetic expressions of the following algebraic expressions.

$$\frac{1}{c} + \frac{1-a}{1+b} \qquad 1/c + (1-a) / (1+b)$$

$$\frac{-b+a^2}{2a+1} \qquad (-b+a*a) / (2*a+1)$$

Q2. (20 pts) Write a C program that finds the largest in a series of numbers entered by the user. The program must prompt the user to enter numbers one by one. When the user enters 0 or negative number, the program must display the largest entered nonnegative number and the number of entered numbers. A sample output of the program is given as follows:

```
Enter a number: 60
Enter a number: 38.3
Enter a number: 4.78
Enter a number 105.62
Enter a number: 70.2295
```

```
The largest number entered was 105.62
The number of entered number is 5
```

Notice that the numbers aren't necessarily integers.

```
#include <stdio.h>
int main()
{
    double number, max;
    int counter=0;
    max = 0;
    printf("Enter a number:");
    scanf("%lf", &number);
    while(number>0)
    {
        counter++;

        if (number > max)
            max = number;

        printf("Enter a number:");
        scanf("%lf", &number);
    }

    printf("The largest number entered was %lf\n", max);
    printf("The number of entered numbers is %d\n",
counter);
    return 0;
}
```

Q3. (20 pts) Write necessary C statements for the following descriptions. If necessary, do necessary declarations and initialization in your solution.

- a) Declare a one-dimensional `int` array `arr` of length 10 and initialize all elements to 1.

```
int arr[10];
int i;
for(i=0; i<10; i++) arr[i] = 1;
```

- b) Assign 10 and 40 to the first and fifth element of the array `arr` in part (a) respectively and display these elements on the screen.

```
arr[0] = 10;
arr[4] = 40;
printf("first value %d fifth value %d\n",
      arr[0], arr[4]);
```

- c) Enter new values for the array `arr` defined in part (a) from the keyboard and print the array elements in one row on the screen.

```
int i;

for(i=0; i<10; i++)
{
    scanf("%d", &arr[i]);
    printf("%d ", arr[i]);
}
```

- d) Find the summation of array `arr` elements defined in part (c) and print the result on the screen.

```
int sum = 0;
for(i=0; i<10; i++)
    sum += arr[i];
printf("summation %d \n", sum);
```

Q4. (16 pts) Trace the following C program and write the output into the corresponding boxes given below.

Note that;

- *Four numbers will be displayed in the output and while you are writing the output, you must fill the given boxes below according to the display order of these numbers.*
- *Every box must contain only one number.*

```
#include <stdio.h>
int main()
{
    int R=0;

    do
    {
        switch (R)
        {
            case 0:
                printf("%d\n",R);
            case 1:
                R += 1;
                break;
            case 2:
                printf("2\n");
            case 3:
                printf("%d\n",R * 2);
            case 4:
                R = R + 3;
                break;
            default:
                R = R * 3;
                printf("%d\n",R / 2);
        }
    }
    while(R<=10);

    return 0;
}
```

0
2
4
7

Q5. (14 pts) The aim of the following C program is to initialize 4x4 matrix, change the diagonal elements in the matrix to 20 and display the average of all elements on the first and last columns of the matrix. Now, complete the missing parts in the following C program by considering both the comments given in the program and the aim.

Note that;

- *In the program, the array "A" must be used as your 4x4 matrix.*
- *You must NOT use additional variables.*
- *You must fill only the missing parts.*

```
#include<stdio.h>
#define L 4
int main()
{
    // In the following statement, the array "A" is declared as 4X4 matrix
// and initialized as 4x4 matrix.
    int A[L][L]={1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16};

    //Variables are declared and one of them is also initialized.
    int i;
    int j;
    int sum=0;

    // The following loop assigns 20 to the diagonal elements.
    for ( i=0 ; i<L ; i++ )

        A[i][i]=20 ;

    // The following nested loops find the sum of all elements
// on the first and last columns of the array "A".
    for( i=0 ; i<L ; i++ )

        for( j=0 ; j<L ; j=j+3 )

            sum= sum + A[i][j] ;

    // In the following statement, the average will be displayed.
    printf("Average of the first and last columns is %f\n", (double)sum / 8);

    return 0;
}
```

Q6. (20 pts) In the following program, the arrays “A” and “B” are declared and initialized. Furthermore, the array “Max” is declared.

Now, complete the following program to compare the first element of the array “A” with the first element of the array “B” and write the maximum element into the first element of the array “Max”, compare the second element of the array “A” with the second element of the array “B” and write the maximum element into the second element of the array “Max”, compare the third element of the array “A” with the third element of the array “B” and write the maximum element into the third element of the array “Max” and so on. The process of comparison and storing maximum element in “Max” array must be done for every element in the arrays.

*Note that you **must** use **loop-structure** to do aforementioned task.*

```
#include<stdio.h>
int main()
{

    int A[8] = { 10 , 22 , 15 , 31 , 25 , 62 , 44 , 36 };
    int B[8] = { 47 , 15 , 81 , 72 , 16 , 29 , 41 , 33 };

    int Max[8];

    int i;

    for ( i=0 ; i<8 ; i++ )
    {

        if ( A[i] > B[i] )
            Max[i] = A[i] ;
        else
            Max[i] = B[i] ;

    }

    return 0;
}
```

Q7. (10 points) What is the output of the following program?

```
#include <stdio.h>

int foo(int n, int k) {
    int i;
    for (i = 1; i < k; i++)
        n /= 10;
    return n % 10;
}

int main() {
    printf("%d\n", foo(359, 1));
    printf("%d\n", foo(359, 2));
    printf("%d\n", foo(359, 3));
    printf("%d\n", foo(359, 4));
    return 0;
}
```

OUTPUT

```
9
5
3
0
```

Q8. (15 points) The following program calculates the median of three integer numbers entered by the user. Complete the missing parts of the program.

```
#include <stdio.h>
```

```
int median(int x, int y, int z) {
    int result;
    if (x <= y)
        if ( y<=z ) result = y;
        else if ( x<=z ) result = z;
        else result = x;
    else {
        if (z <= y) result = y;
        else if ( x<=z ) result = x;
        else result = z;
    }
    return result;
}
```

```
int main()
{
    int x, y, z;

    printf("Enter three numbers: ");
    scanf("%d%d%d", &x, &y, &z);

    printf("Median of %d, %d, and %d is %d\n", x, y, z, median(x, y, z));
    return 0;
}
```

SAMPLE OUTPUTS:

```
Enter three numbers: 45 27 39
Median of 45, 27, and 39 is 39
```

```
Enter three numbers: 67 145 44
Median of 67, 145, and 44 is 67
```

```
Enter three numbers: 15 25 35
Median of 15, 25, and 35 is 25
```

Q8.(10 points) Trace and find the output of the following program. Write your output in the box provided below. Please enter one character in every box.

```

#include <stdio.h>

#define N 4

int main (void)
{
    int i,j;

    int A[N]={1,2,3,4};

    int B[N]={1,-1,1,-1};

    int C[N];

    int D[N][N]={{1,1,1,1},{1,1,1,1}};

    for (i=0; i<N; i++)
    {
        if ( i %2 ==1)
            C[i]=A[i]*B[i];

        else
            C[i]=A[i]+B[i];

        D[i][i]=C[i];
    }

    for (i=0; i<N; i++)
    {
        printf ("%d\n", C[i]);

        for (j=0; j<N; j++)
            printf ("%d", D[i][j]);

        printf ("\n");
    }

    return 0;
}

```

OUTPUT

2			
2	1	1	1
-2			
1	-2	1	1
4			
0	0	4	0
-4			
0	0	0	-4

Q9. (15 points) The dot product of two vectors $\mathbf{a} = [a_1, a_2, \dots, a_n]$ and $\mathbf{b} = [b_1, b_2, \dots, b_n]$ is defined as.

$$\mathbf{a} \cdot \mathbf{b} = \sum_{i=1}^n a_i b_i = a_1 b_1 + a_2 b_2 + \dots + a_n b_n$$

where \sum denotes summation notation and n is the dimension of the vector space.

For instance, in three-dimensional space, the dot product of vectors $\mathbf{a}=[1, 2, -6]$ and $\mathbf{b}=[4, 2, -1]$ is:

$$[1,2,-6] \cdot [4,2,-1] = (1)(4) + (2)(2) + (-6)(-1) = 14$$

Fill in the missing parts in the following program such :

- that the user will enter two 5 dimensional vectors \mathbf{a} and \mathbf{b} into the arrays A and B respectively
- the dot product of the two vectors will be calculated and printed as the output.

Note: Use only the spaces provided. Do not add any extra lines or delete any given code.

```
#include <stdio.h>

int main (void)
{
int i, sum=0;
/*declare your arrays A and B below*/
int ____A[5]_____, ____B[5]_____
/* Use the loop below to enter elements of the two arrays */
for ( _i=0; i<5;i++ _____)
    ____scanf ("%d%d", &A[i],
&B[i])_____ ;
/* Use the loop below to calculate the dot product here */
for ( _____ i=0; i<5;i++ _____)
    ____sum+=A[i]*B[i]_____ ;
/* print the dot product below */
____printf("%d", sum)_____ ;
return 0;
}
```

Question 1-) (22 pts.) Trace the following C program and write the output into the corresponding box, which is divided into cells, below. Assume that you have 4-cell-to-8-cell display screen and each cell shows one character on the screen. *Note: if any character is not printed, let the corresponding cell empty.*

```
#include <stdio.h>

int main()
{
    int n=10;
    int j=10;

    for ( n=7 ; n>=1 ; n=n-2 )
    {
        for ( j=0 ; j<n ; j++ )
            printf( "%d" , n-1 );
        printf( "C\n" );
    }

    return 0;
}
```

OUTPUT:

	1	2	3	4	5	6	7	8
1	6	6	6	6	6	6	6	C
2	4	4	4	4	4	C		
3	2	2	2	C				
4	0	C						

Question 2-) (22 pts.) Trace the following C program and write the output into the corresponding box below.

```
#include <stdio.h>
int main()
{
    int Number=0;

    while (Number<5)
    {
        switch (Number)
        {
            default:
                printf("Magusa\n");
            case 0:
                printf("Girne\n");
                break;
            case 3:
                printf("Lefkosa\n");
                break;
            case 2:
                ++Number;
            case -1:
                printf("Iskele\n");
                break;
            case 1:
                printf("Karpaz\n");
        }
        ++Number;
    }

    return 0;
}
```

OUTPUT:

**Girne
Karpaz
Iskele
Magusa
Girne**

Question 3-) (32 pts.) Complete the missing parts in the following C program to produce the output below. Be aware of odd numbers will be displayed one time and even numbers will be displayed according to its value. For example; 1 is an odd number and displayed one time; 2 is an even number and displayed 2 times; 3 is an odd number and displayed one time; 4 is an even number and displayed 4 times; etc.

Notes: Do not use any variable other than the variables declared in the program.

OUTPUT:

```
1 2 2 3 4 4 4 4 5 6 6 6 6 6 6 7 8 8 8 8 8 8 8 8
```

```
#include <stdio.h>

int main()
{
    int N=8;
    int i=1;
    int j;

    while (i <= N)
    {
        if (i % 2 == 0)
        {
            for( j=1 ; j <= i ; j++)
                printf( "%d " , i );
        }
        else
            printf( "%d " , i );
        i++;
    }

    return 0;
}
```

Question 4-) (32 pts.) Complete the following C program to search the number "**number**", given by the user, in the elements of the array "**A**". If it finds the number, it will give the message "**It is found!!**". If it does not find the number, it will give the message "**It is NOT found!!**". Some sample outputs are given below.

Note: If you need, you can use additional variable(s).

```
#include <stdio.h>
#define L 15

int main()
{
    int A[L]={1,4,6,9,23,2,5,91,21,45,67,54,44,88,27};
    int number;

    printf("Enter a number to search:");
    scanf("%d",&number);

    int i;
    int check=0;

    for ( i=0 ; i<L ; i++ )
    {
        if (number==A[i])
        {
            check=1;
            break;
        }
    }

    if (check==1)
        printf("It is found!!\n");
    else
        printf("It is NOT found!!\n");

    return 0;
}
```

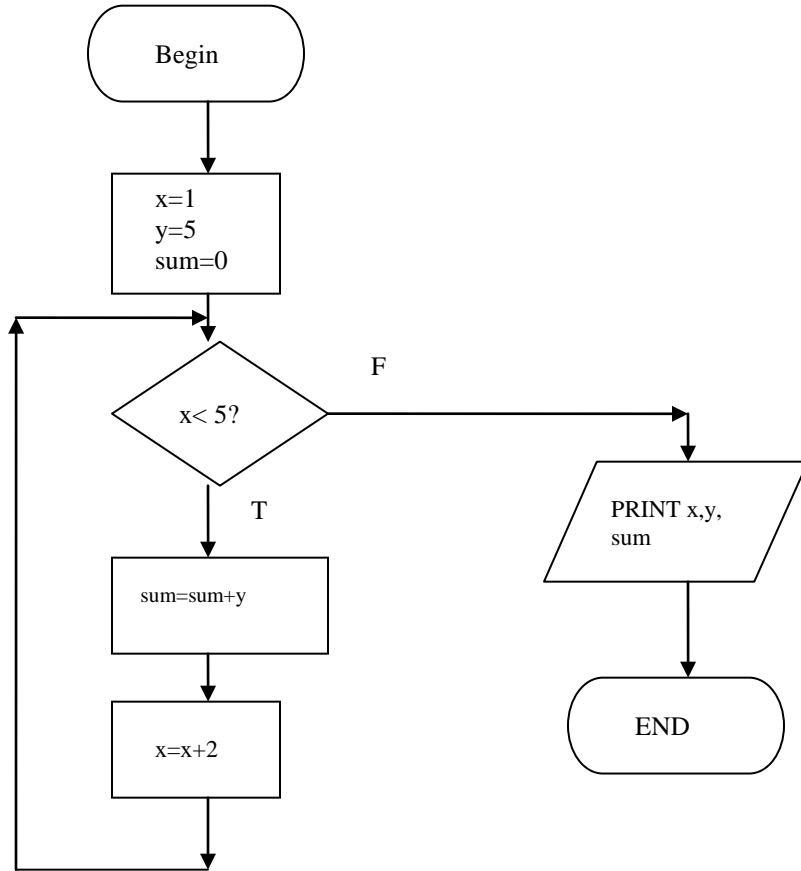
```
C:\ "C:\Program Files\Microsoft Vis
Enter a number to search:18
It is NOT found!!
Press any key to continue_
```

```
C:\ "C:\Program Files\Microsoft Vis
Enter a number to search:11
It is NOT found!!
Press any key to continue_
```

```
C:\ "C:\Program Files\Microsoft Vis
Enter a number to search:5
It is found!!
Press any key to continue_
```

QUESTIONS WITHOUT SOLUTIONS (for you to test yourself)

Q1) Write a C code which is equivalent to the flowchart shown below. (15 points)



ANSWER (your code)

Q2) a) Evaluate the following expressions and write your answers in the boxes provided.

(7 points)

- | | ANS |
|---------------------------|----------------------|
| i) $10.0+15/2*6.0$ | <input type="text"/> |
| ii) $3*4.0/6+6$ | <input type="text"/> |
| iii) $10+17/3.+4$ | <input type="text"/> |
| iv) $3.0*4\%6+6$ | <input type="text"/> |
| v) $10+17\%3+4.0$ | <input type="text"/> |
| vi) $2*5==4*4$ | <input type="text"/> |
| vii) $5\%2*4>5 4\%2*5<7$ | <input type="text"/> |

b) Write relational expressions in C to express each of the following conditions. (10 points)

(example: x is equal to 5 \rightarrow $x==5$)

- | | ANS |
|--|----------------------|
| i) y is greater than or equal to 6 | <input type="text"/> |
| ii) y is equal to x and greater than z | <input type="text"/> |
| iii) x is positive but not equal to 4 | <input type="text"/> |
| iv) x is not equal to 5 or -5 | <input type="text"/> |
| v) ch is between 'd' and 'm' | <input type="text"/> |

c) Write an equivalent C expression for the following mathematical formulas. (5 points)


i) $a + \frac{bc}{x + \frac{2}{y}}$

ii) $-x^3 + 3x^2 + 5x - 6$

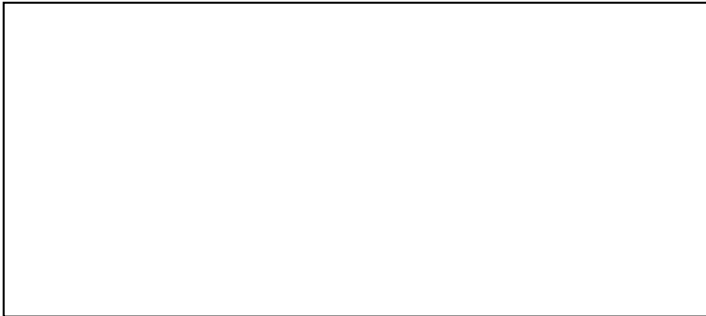
Q3) Consider the *while loop* below:

```
i=10;
while ( i >= 1)
{
    printf ("ok\n");
    i=i-2;
}
```

a) Rewrite the code above using a *for loop*. (5 points)



b) Rewrite the code above using a *do while loop*. (5 points)



c) Rewrite the following code using *while loops*. (10 points)

```
for ( i=1; i<10; i++)
    for (j=3; j>0; j--)
        printf ("*\n");
```

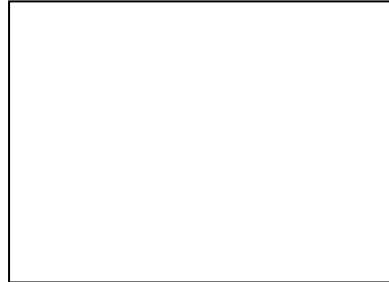


Q4) Write the output of each of the following codes below to the boxes provided.
(Only the answer box will be graded-no partial credit points for work done!)

a) (5 points)

```
for (i=1;i<=5;i++)
{
    for(j=1;j<=i;j++)
        printf("*");
    printf("\n");
}
```

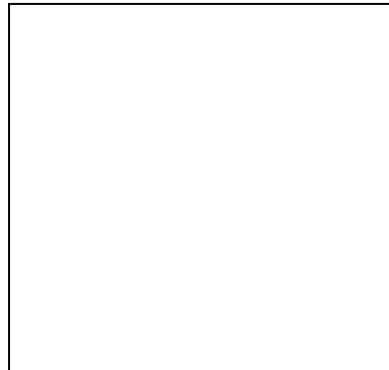
ANSWER



b) (5 points)

```
i=5;
while(i>=1)
{
    j=1;
    while (j<=i)
    {
        printf("*");
        j++;
    }
    printf("\n");
    i--;
}
```

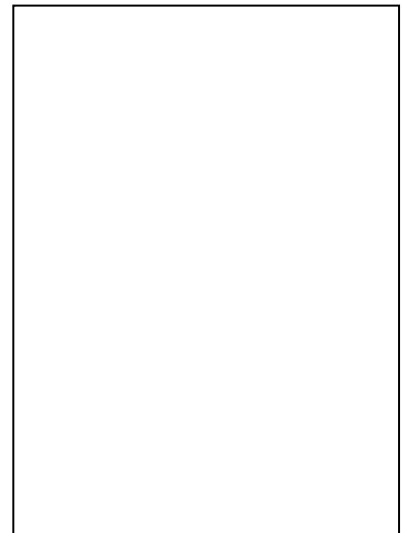
ANSWER



c) (8 points)

```
j = 0;
do
{
    j++;
    switch(j)
    {
        case 2: printf("C\n");
        case 1: printf("B\n"); break;
        case 3: printf("D\n");
        default: printf("E\n");
    }
    printf("X\n");
}while(j < 6);
```

ANSWER



Q5) Consider the following C program whose description and functionality is given below:
 The program calculates the tax pay for a 100 employees. The tax pay for each employee depends on the age and marital status ($m_status=1$ for single and $m_status=2$ for married) of the employee. Data (age, salary and marital status) is entered by an operator who must have a valid security code between 1 and 9999. The operator cannot enter data if his security code is not valid and is asked to re-enter his code every time he enters it wrongly. There are 4 different tax rates, rate1, rate2, rate3, and rate4. The tax rate which of an employee is determined using the following table and the tax to be paid is given by: $tax=salary*tax\ rate$

Age	Marital status	Tax rate
<30	single	rate3
<30	married	rate4
≥30	single	rate1
≥30	married	rate2

Complete the missing parts of the code such that the program functions correctly (25 points)

```
#include <stdio.h>

#define rate1 0.05
#define rate2 0.03
#define rate3 0.01
#define rate4 0

main() {
int age, salary, emp_no, m_status, sec_code;
float tax;

do{
printf ("\n Please enter security code");
scanf ("%d", &sec_code);
} while (_____ )

emp_no_1;
while (_____ ) {
printf ("\n enter age, salary and marital status");
scanf( _____ );

if (_____ )
tax=salary*rate1;
else if (_____ )
if (_____ )
tax=salary*rate2;
else
tax=_____ ;
else
tax=_____ ;

printf ("\n Tax pay for employee %d = %.2f", emp_no, tax);
}
}
```

Q6) Given the definition of the function

```
float func(int a, int b)
{
    if (b<5) return 1.0*a/b;
    else return a%10;
}
```

and the integer variables

```
int Total=8, Account=1, Sum=3, GNP=1, b=16;
```

evaluate the following C expressions,

- a) `3.0*func(2,Total)`
- b) `func(b,b)/func(b,Sum+1)`
- c) `Total - Sum*Sum >= Total*GNP - b*b`
- d) `Total > 3 == Sum`
- e) `Total != b - b < 3`

Q7) Compute the value of m for the following C fragments.

- a) `int j = 2, k = 6, m;`
`m = ++j + k--;` -----
- b) `int j = 2, k = 3, m;`
`m = j + (1 > k-j);` -----
- c) `int i = 1, m;`
`m = !i != 5;` -----
- d) `int i = 1, j = 3, m;`
`m = (I < j) % j;` -----
- e) `int i = 1, j = 1, m;`
`m = ++i && --j;` -----
- f) `int j = 1, k = 6, m;`
`m = --j ? --k : ++k;` -----