



Eastern Mediterranean University  
Department of Computer Engineering

CMPE 112 Final Exam  
Fall Semester 2019-2020  
January 8, 2020

Name, Surname :...**SOLUTIONS**.....  
Student No :.....  
Group No :.....

**Instructors:** Prof. Dr. M. Güler (Gr. 01), Prof. Dr. E. Varoğlu (Gr. 02)

**Duration: 100 minutes**

**Instructions:**

- There are **5** questions in this **7** page sheet.  
Please check !!!!!
- Calculators are not allowed.
- Mobile phones must be turned off.
- A table of operators for precedence and associativity is attached.
- Passing any material including rubbers, pencils etc. to anybody else is strictly prohibited in the exam.
- If an answer box is given in a question, you must give your answer (and nothing else) in the box !!!

Question	Grade
Q1 (16 pts)	
Q2 (10 pts)	
Q3 (48 pts)	
Q4 (10 pts)	
Q5 (16 pts)	
<b>TOTAL:</b> (out of 100)	

## PRECEDENCE AND ASSOCIATIVITY

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

### Some String Functions:

**strlen(s1)** Returns length of the string s1

**strcat( s1 , s2 )** Concatenates a copy of string s2 onto the string s1

**strncat(s1, s2, n)** Concatenates a copy of up to n characters from string s2 onto the string in s1

**strcpy(s1, s2)** Copies the string s2 to s1

**strncpy(s1, s2, n)** Copies a string up to n characters from s2 to s1

**strcmp(s1,s2)** Compares the string s2 with the string s1. Returns an integer less than, equal to, or greater than depending on the result of the comparison

**strncmp(s1, s2, n)** Compares at most the first n characters of the string s1 to s2, and returns an integer less than, equal to , or greater than depending on the result of the comparison

**strchr(s1 , c)** Locates the first occurrence of c(a char) in the string s1, and returns a pointer to the located character. If the search is not successful than null is returned

**strrchr(s1, c)** Locates the last occurrence of c(a char) in the string s1, and returns a pointer to the located character. If the search is not successful than null is returned

**Q1)(16 points)**

Give the output of the following program :

```
#include <stdio.h>
void group(char a, int c[]);
void main(void)
{
    char one , line[]="software design issues";
    int cnt[6]={0};
    int i=0;

    do
    {
        one=line[i++];
        group(one,cnt);
    }while (line[i] != '\0');

    for(i=0;i<6;i++)
        if (cnt[i]>0) printf("%d %d \n",i+1,cnt[i]);
}

void group(char a, int c[])
{
    int i;
    switch (a)
    {
        case 'a': i=1; break;
        case 'e': case 'i': i=2; break;
        case 'o': case 'u': i=3; break;
        default : i=0;
    }
    ++c[i];
}
```

1	14
2	1
3	5
4	2

**Q2)(10 points)**

Give the output of the following program :

```
#include <stdio.h>
main()
{
    char s[] = "COMPUTER_SCIENCE", news[10];
    int i=0, j=0;
    while(s[i++] != '\0')
        if(s[i] == 'E') news[j++] = s[i-1];
    news[j] = '\0';
    printf("%s\n", news);
}
```

TIC

**Q3)(48 points)**

Give the output for each of the following C programs:

a)

```
#include <stdio.h>
void fun(int *p, int x)
{
    *p += 4;
    x += 4;
}
```

```
main()
{
    int a = 4, b = 6;
    fun(&a, b);
    printf("a = %d b = %d\n", a, b);
}
```

---

**a = 8 b = 6**

b)

```
#include <stdio.h>
main()
{
    int k, *p, v[] = {1, 2, 3, 4, 5};
    p = v + 2;
    *p = 8;
    p[2] = 12;
    p[-1] = 21;
    v[0] = p[1] + 10;
    *(v+3) = *p + 2;
    for(k = 0; k <= 4; k++) printf("%d ", v[k]);
}
```

---

**14 21 8 10 12**

c)

```
#include <stdio.h>
main()
{
    int x, *p, v[] = {2, 5, 3, 4, 6};
    p = v;
    x = ++*p + 7;
    printf("%d\n", x);
    x = *p++ + 7;
    printf("%d\n", x);
    printf("%d %d\n", p[2], v[0]);
}
```

---

**10  
10  
4 3**

d) `#include <stdio.h>`  
`void fun(int *p, int size)`  
`{`  
`int k;`  
`for(k=0; k<size; k++) p[k] += k+10;`  
`}`

**2 15 14 16 19 9**

`main()`  
`{`  
`int k, v[] = {2, 5, 3, 4, 6, 9};`  
`fun(v+1, 4);`  
`for(k=0; k<= 5; k++) printf("%d ", v[k]);`  
`}`

---

e) `#include <stdio.h>`  
`int *fun(int *p, int a)`  
`{`  
`int *q;`  
`q = p + a;`  
`return q;`  
`}`

**4**

`main()`  
`{`  
`int *fun(int *, int);`  
`int *r, v[] = {1, 2, 3, 4, 5, 6};`  
`r = fun(v+1, v[1]);`  
`printf("%d\n", *r);`  
`}`

---

f) `#include <stdio.h>`  
`#include <string.h>`  
`void main()`  
`{`  
`int sing(char *);`  
`char s[] = "Searching and singing with joy";`  
`printf("Result = %d", sing(s));`  
`}`

**Result = 2**

//continues on the next page

```

int sing(char s[])
{
    int i, cnt=0;
    char tree[4];

    for(i=0;i<strlen(s);i++)
    {
        strncpy(tree,s+i,4);

        //Note the blank in "ing "
        if(strncmp("ing ", tree,4) == 0) cnt++;
    }
    return cnt;
}

```

#### Q4)(10 points)

The following program reads a string from the keyboard until EOF(ctrl + z) is pressed. In this string, there may be any valid character (example string: “Mt-Exam2\$+#”).

The program puts the *uppercase letters* into the array named as **one**.

For the string above, we have **one** = “ME”. It then displays the array. Complete the missing parts of the following program.

```

#include <stdio.h>
main()
{
    int c, one_index=0;
    char one[10];

    while(_____(c=getchar()) != EOF_____)

        if(_____c >= 'A' && c<= 'Z'_____) one[_____one_index++_____] = c;

    one[one_index] = '\0';
    puts(one);
}

```

#### Q5)(16 points)

The program below declares a 10 x 10 matrix and

- 1) Enables the user to fill in the matrix elements
- 2) Finds the minimum (min) and maximum (max) values of the matrix elements
- 3) Calculates the number of zeros in the matrix
- 4) Finds the sum of the diagonal elements of the matrix.

//continues on the next page

Notes :

- a. Diagonal elements of an  $n \times n$  matrix  $A$  are  $A_{kk}$ ;  $k= 1,2,...,n$
- b. Assume that the values entered are in the range  $[1,10000]$

```
#include<stdio.h>
int main(void)
{
    float matrix[10][10], number;
    int i, j, zero_count=0, diag_sum=0, min=10000, max=0;

    for (i=0;i<10;i++)
        for (j=0;j<10;j++)
        {
            scanf("%f", &number);
            matrix[i][j] = number;
        }

    for (i=0;i<10;i++)
        for (j=0;j<10;j++)
        {
            if( i == j) _____diag_sum += matrix[i][i] _____;

            if(matrix[i][j] == 0) _____zero_count++_____;

            if(matrix[i][j] > max) _____max = matrix[i][j] _____;

            if(matrix[i][j] < min) _____min = matrix[i][j] _____;
        }

    printf("diag_sum =%d, zero_count =%d, max=%d, min=%d\n",
           diag_sum, zero_count, max, min);

    return 0;
}
```

END OF EXAM