KSU/CCIS/CS	**CSC 215**	**Final Exam- Fall 13-14** **2 Hours**
Name:	**ID: ....**	

EXERCISE 1 **(/20pts)**

Write True/ False

In C, memory management is left to the programmer.	
In a binary operation, the conversion of the “lower” type operand to the “higher” type operand is done automatically.	
C allows a function to be defined inside another function.	
If a local variable and a global one have identical names, all references to the name within the function will refer to the global variable.	
An array cannot be copied into another array by assigning it to that array.	
In C, Arrays can have more than one dimension.	
strdup copies a string into a newly created location	
The value returned by **isalnum(‘9’)** is 0	
Two member variables in different structures can have the same name	
The unary operators & and * have the same precedence as any other unary operator, with associativity from right to left.	

EXERCISE 2 **(/20pts)**

Select the correct answer

When a continue statement is encountered within a loop body,

a) The execution of the loop body is interrupted, and the program control transfers to the exit point of the loop.
b) All the remaining statements in the loop body are skipped and the loop continuation condition is evaluated next.
c) The program stops.
d) Nothing happens.

When a function calls itself (directly, or indirectly) it is called a

a) Self
b) Recursive
c) Referring
d) None of the above

Which of the following is **NOT** an error when using free:

a) Use free on a NULL pointer
b) Use free on a pointer that has already been freed
c) Use free on a memory address directly returned by malloc.
d) Use free on a memory address that has been statically allocated

Which of the following cannot be a structure member?

a) Another structure
b) Function
c) Array
d) None of the mentioned

If **b** is a pointer to a structure, which of the following accesses its member variable **var**.

a) b->var;
b) b.var;
c) b-var;
d) b>var;

EXERCISE 3 **(/20pts)**

Write the output of the following C program.

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

  int a = 10 , b=9,c=8;

  printf(“a > b: %d\n”, a>b);

  printf(“a-c==b+c : %d\n”, a-c==b+c);

  printf(“a+=b!=c: %d\n” , a+=b!=c);
return 0;
}
```

Write the output of the following C program.

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

  int i, n=21, sum=0;
  for (i = 1; i <= n; i++) {
       if (i % 3 != 0) { continue; }
       sum += i;
  }
  printf(“The value of sum is %d\n”, sum);

  return 0;
}
```

Write the output of the corresponding C program

```
#include <stdio.h>

void printB (int num) {

   printf("%d\n", num);

   if (num > 1)
      printB(num - 1);
}
main()
{
   printB(4);
}
```

Write the output of the following C program

```
     #include <stdio.h>
     int main()
     {
       char s[100] = "riyadh";
       char *p1 = &s[0];
       printf("The value of *p1 is %c\n", *p1);
       char *p2 = &s[4];
       printf("The value of p2-p1 is %d\n", p2 - p1);
       return 0;
     }
```

EXERCISE 4 (/20pts)

Write the code to create an array called **f** of 8 floats and dynamically allocate the memory to the elements of the array and initialize the values of **f** to zero.

Declare a structure called **person** with the following elements: name as character array of size 20, age as an integer and gender as character.

Declare a structure variable called student of type **person** (from previous question) with initial values: name “Ahmed”, age 20 and gender ‘M’

Write the function **isupper** as defined in ctype library. You may **NOT** use any other function from the ctype library.

```
int isupper(int c){





}
```

EXERCISE 5 **(/20pts)**

Write a C program that implements the following requirements:

1- A recursive function called **factorial** that takes an integer n computes the factorial of n

2- A function called **square** that takes an array of doubles and the size of the array as arguments and replaces each array element with its square.
(Example: (1.0,4,6) and size 3 replaces the array values with (1.0,16.0,36.0))

3- A **main** function with the following requirements:
   a. Ask the user to enter a number **n**.
   b. Compute and print the factorial of **n**.
   c. Declare a double array of size **n** called **a**.
   d. Ask the user to enter numbers and save them in **a**.
   e. Replace the values of **a** by their square values.
   f. Print the new values of **a**, one value per line.