C Programming Language: mid-term exam (180 minutes)

1	2	3	4	5	6	7	8	9	10	11	12	13	Total
5	10	5	10	10	10	10	10	15	10	10	15	10	130

1. [5] Given a 2D integer array A of size nxn, write a function that computes the sum of the maximum numbers in all the rows. Also, write the main function to test it.

2. **[10]** Is the following program segment legal in C? If legal, what is the result? Explain.
   (a) int x=0; if (x = 1) printf("%d\n", x);
   (b) x = (3<2) ? 3 : 2>1 ? 2 : 1; printf("%d\n", x);
   (c) x = 1; printf("%d\n", x++); printf("%d\n", ++x);
   (d) for(x=2;x=1;x=0) printf("%d\n", x);
   (e) i = 0;
   while (i <10){ i++; break; } printf("%d\n", i);

3. **[5]** Explain what is "Call-by-value". Please give an example.

4. **[10]** The maximum and minimum values for a variable of type *int* are INT_MAX= 2147483647(2^{31}-1) and INT_MIN = -2147483648(-2^{31}), respectively.
   Which header file defines these two constants?
   Write a function to check if the product of two integers is between -2147483648 and 2147483647 inclusively and no overflow occurs. Please design a new format by using two consecutive integers to store a number that is between -2^{63} and 2^{63}-1. Write a function *product(int a, int b, int result[])* that computes the product of numbers a and b and stores the results in array of two integers.

5. [10] Write a function that cuts the integer interval [***begin***, ***end***] into many sub-intervals based on the following two rules: (1) each sub-interval must contains n number, where *n* is a power of 2 and (2) the numbers belonging to a sub-interval must be consecutive and have the same prefix. For example, the interval [3, 10] will be cut into sub-intervals, [3, 3], [4, 7], [8, 9], and [10, 10] because [4, 7] = {0100, 0101, 0110, 0111} has the same prefix '01', [8, 9] = {1000, 1001} has the same prefix 100. [3,4] = {0011, 0100} cannot form a sub-interval because 001 and their prefixes 010 are not the same.

6. [10] Greatest common divisor (gcd) can be defined by the following equations:
   gcd(a, 0) = a and gcd(a, b) = gcd(b, a%b).
   Please write a recursive function and another non-recursive function to compute gcd of two integers

7. [10] Consider the following "mystery" function:

```
void pb(int n)
{
  if (n != 0){
     pb(n/2);
     putchar('0' + n%2);}
}
```

   Explain what this function does by tracing the function by hand using an example.
   Also, write a non-recursive function for the same task.

8. [10] What does the following program output:
   int i, j = 0;

```
for (i=0; i<8; i++){
 switch(i) {
        case 1: j=1;
        case 2: j=2;     break;
        case 3: j=3;     break;
        case 4: (j=4 ? j=9:j=8);   break;
        case 5: if (j=4) j = 7;
              break;
        case 6: j=6;
        default: j=7;
 }
 printf("i = %d, j = %d\n", i, j);
}
```

9. [15] Write a function called void ***month_year_calendar(int year, int month)*** that can print out the calendar of each month of year from 2001 to 2020. For example, when you call ***month_year_calendar(2009,11),*** the following calendar will be printed out on the screen: (note that 1, 2, 3, …, 12 represent January, February, March, …, December and 2009/01/01 is Thursday)

Sun	Mon	Tue	Wed	Thu	Fri	Sat
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30					

10. [10] Use the random number generator to write a function ***myrandom(min, max, p, n)*** that can generate *n* random numbers in the range between ***min*** and ***max*** inclusively, where n, min and max are of type integer and p is a pointer pointing to an existing array of size n. Also, write a complete main program to test the function ***myrandom().***
11. [10] Write a function ***align***(int **n**) that asks the user to enter a paragraph of text including space character, comma (,), and period (.) and outputs an aligned text of **n** characters per line. You have to do the text alignment (both left and right adjusted) by using spaces between words evenly. Note that you need to count the number of characters in each word.

12. **[15] Write a function *PascalTriangle(int n)* that uses a 1D array to store the Pascal's triangle.**

Example:

```
                    1.....(First line)
                  1   1.....(Second line)
                1   2   1.....(Third line)
              1   3   3   1
            1   4   6   4   1
          1   5   10  10  5   1
        1   6   15  20  15  6   1
      1   7   21  35  35  21  7   1
```

By callling ***PascalTriangle(5)***, the 1D array should store the following content:
1 1 1 1 2 1 1 3 3 1 1 4 6 4 1 1 5 10 10 5 1
Write another function independent of ***PascalTriangle(int n)***as follows: The function returns the value based on the row and col (i.e., the ***colth*** number in the ***rowth*** row).
int Pascal(int row, int col)

{...}
Ex: Pascal(5, 3), it returns 6.

13. **[10]** Write a recursive program ***int combinations(A, n, k)*** that you can print out all the combinations of *k* numbers out of *n* numbers stored in an array A with an additional rule: **the sequence of these *k* numbers must be in an decreasing order**. For example, assume there are 4 numbers 4, 1, 2, 3 stored in array A[4]. Calling this recursive function *combinations*(A, 4, 2) can return a value 3 and print out (1, 2), (1, 3), and (2, 3), or calling *combinations*(A, 4, 3) can return a value 1 and print out (1, 2, 3). Your recursive program must consider to avoid the unnecessary recursive function calls and also don't first generate all the combinations and then check if each combination is in an decreasing order.