Problem Solving mid-term exam

1	2	3	4	5	6(a)	6(b)	7	Total

학번	
이름	

1. 다음은 회문(Palindrome)임을 확인하고 출력하는 프로그램의 소스 전문이다. 빈 상자 안에 알맞은 소스를 입력하시오. (10점)

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


int main(void) {
	char input[16];
	int  i, j, length, ret = 1;

	printf("Input string: ");

	fgets( input, sizeof(input), stdin );

	length = strlen(input) - 1;
	i=0;
	j=length-1;


	while (  i < length / 2 ) {
		if (input[i] != input[j]) {
			ret = 0;
			break;
		}
		i++;
		j--;
	}

	if(ret == 1)  puts("Palindrome!");
	else	puts("Normal word!");

	return 0;
}
```

2. 다음 소스 전문과, 주어진 input 에 대한 출력을 아래 빈칸에 작성하시오. (10 점)

소스 전문

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

int main(void){
        char input[16];
        int i;
        int length=0;

        scanf("%s", input);

        for(i=0; i<16; i++){
                if(input[i]=='\0')
                        break;
                length++;
        }

        for(i=0; i<length; i++){
                printf("%c", input[length-i-1]);
        }
        printf("\n");

        return 0;
}
```

주어진 Input

```
abcde
```

출력

```
edcba
```

3. 다음 배열에 insertion sort 방법을 적용하여 정렬되는 과정을 단계별로 나타내시오. (10점)

9	3	7	4	2	6
3	9	7	4	2	6
3	7	9	4	2	6
3	4	7	9	2	6
2	3	4	7	9	6
2	3	4	6	7	9

4. 다음은 quick sort 를 구현한 소스코드이다. 괄호안에 알맞은 코드를 제시된 보기 중에서 골라 채우시오. (20 점)

```
void quickSort(int *parr, int left, int right){
        int q;
        if( (ⓐ) ){
                q = partition(parr, left, right);
                quickSort(parr, left, (ⓑ) );
                quickSort(parr, (ⓒ) , right);
        }
}
int partition(int *parr, int left, int right){
        int pivot = parr[right];
        int i = left - 1;
        int j;
        for(j = left ; j < right ; j++){
                if( (ⓓ) ){
                        i++;
                        swap(&parr[i], &parr[j]);
```

```
                }
        }
        swap(&parr[i+1], &parr[right]);
        return i+1;
}
void swap( int *v1, int *v2 ){
        int t = *v1;
        *v1 = *v2;
        *v2 = t;
}
```

<보기>

parr[j] > pivot	q	q - right	q - 1	q + left
q + 1	q < right	left < right	parr[i] < pivot	parr[j] <= pivot

ⓐ	left < right
ⓑ	q – 1
ⓒ	q + 1
ⓓ	parr[j] <= pivot

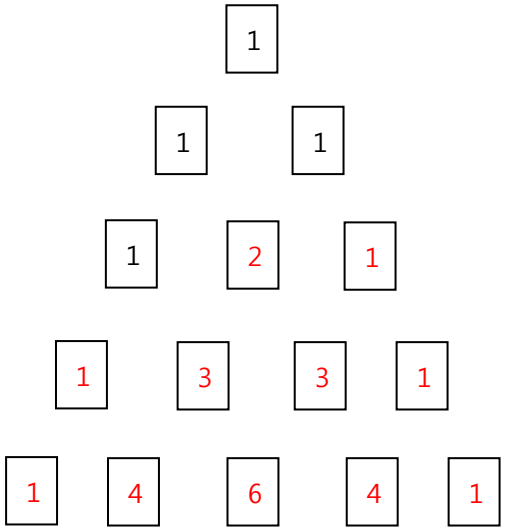
5. C언어에서 integer type으로는 $\pm 2^{31}$(2,147,483,648)의 범위 안에서만 수를 다룰 수 있으므로, 큰 수의 연산을 위해서는 다른 방법을 사용해야 한다. 다음은 100자리 숫자의 크기 비교가 가능한 프로그램이다. (20점)

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

#define MAXDIGIT      100
#define PLUS          1
#define MINUS         -1

typedef struct {
        char digits[MAXDIGIT];        //입력 받은 수 저장
        int signbit;                  //양수면 PLUS(=1), 음수면 MINUS(=-1)
        int lastdigit;                //입력 받은 수의 자릿수
} bignum;

//a 와 b 의 크기 비교 함수
//a가 더 클경우 양수 b가 더 클경우 음수 같을경우 0을 리턴.
int compare_bignum(bignum *a, bignum *b) {
        int i=0;

        if(a->signbit == PLUS && b->signbit == MINUS)
                return PLUS;
        if(a->signbit == MINUS && b->signbit == PLUS)
                return MINUS;
        if(a->lastdigit > b->lastdigit)
                return ( PLUS * a->signbit );

        if(a->lastdigit < b->lastdigit)
                return ( MINUS * a->singbit );

        else{
                for(i=a->lastdigit; i>=0; i--){
                        if(a->digits[i]>b->digits[i])
```

```
                        return PLUS;
                else if(a->digits[i]<b->digits[i])
                        return MINUS;
                }
        }
        return 0;
}

//a->digits 출력 함수
void print_bignum(bignum *a) {
        int i;

        if(a->signbit == MINUS) printf("-");

        for (i = a->lastdigit - 1 ; i >= 0 ; i-- )
                printf("%c", a->digits[i]);
        printf("\n");
}

void main() {
        int i;
        //bignum a, b, c의 동적 메모리 할당
        bignum *a = (bignum*)malloc(sizeof(bignum));

        bignum *b = (bignum*)malloc(sizeof(bignum));

        bignum *c = (bignum*)malloc(sizeof(bignum));

        char tmp[MAXDIGIT];

        //사용자로부터 수 입력 받기
        gets(a->digits);
        gets(b->digits);

        a->lastdigit = strlen(a->digits);
        b->lastdigit = strlen(b->digits);

        strcpy(tmp, a->digits);
```

```
        for(i=0; i<a->lastdigit; i++)
                a->digits[i] = tmp[a->lastdigit - i - 1];

        strcpy(tmp, b->digits);
        for(i=0; i<b->lastdigit; i++)
                b->digits[i] = tmp[b->lastdigit - i - 1];

        //수 배열에서 마이너스 부호 삭제
        if(a->digits[a->lastdigit - 1] == '-') {
                a->signbit = MINUS;
                a->digits[a->lastdigit - 1]=NULL;
                a->lastdigit--;
        }
        else
                a->signbit = PLUS;

        if(b->digits[b->lastdigit - 1] == '-') {
                b->signbit = MINUS;
                b->digits[b->lastdigit - 1]=NULL;
                b->lastdigit--;
        }
        else
                b->signbit = PLUS;

        //큰 수 출력
        printf("The bigger number is ");
        if( compare_bignum(a,b) == PLUS)
                print_bignum(a);
        else
                print_bignum(b);
}
```

6. (a) ${}_{(n}C_k$ 는 k개중에 n개를 고르는 방법의 수를 의미한다. ${}_nC_k$ 를 구하는 방법중에 하나로 Pascal의 삼각형을 그리는 방법이 있다. Pascal의 삼각형의 각 숫자는 그 숫자 윗줄의 두숫자를 합한 값이다. 다음은 파스칼의 삼각형의 예시이다. 빈칸을 채우시오. (5점)



(b) Pascal의 삼각형을 이용해서 ${}_nC_k$ 을 구하는 프로그램을 간단히 작성할 수 있다. 다음은 ${}_nC_k$ 를 구해서 출력해주는 프로그램이다 프로그램의 빈칸을 채우시오. (10점)

```
#include <stdio.h>

// nCk 의 값을 리턴하는 함수

int bc(int n, int k)
{
    if (n==k) return 1 ;
    if (k==1) return n ;
    return bc(n-1, k-1) + bc(n-1,k) ;
}
int main()
{
    int n,m;
    while ( scanf("%d %d", &n, &m) != EOF  )
    {
        printf("%lu\n", bc(n,m));
    }
    return 0;
}
```

7. 다음은 피보나치 수열의 n번째 항을 Recursive 함수로 구하는 프로그램이다 이 프로그램의 출력 결과를 쓰시오. (10점)

```
#include <stdio.h>

int fibonacci(int n)
{
        printf("%d\n", n);
        // base case of recursion
        if(n == 1)
                return 1;
        else if(n == 2)
                return 2;
        else
                return fibonacci(n-1) + fibonacci(n-2);
}

int main(void)
{
        printf("5th Fibonacci number is %d\n", fibonacci(5));
        return 0;
}
```

```
5
4
3
2
1
2
3
2
1
5th Fibonacci number is 8
```