

CODING TEST
An examiner : Son Hyegang The copyright on the problem of the coding test belongs to the problem writer. Do not allow external leakage. The list of references is excluded.

1. 197 은 circular prime 이라고 불리는데, 그 이유는 이 숫자가 prime 이면서 이것을 rotate 한 값, 즉 971, 719 도 역시 prime 이기 때문이다. 어떤 수가 circular prime 인지 확인할 수 있도록 다음 함수를 완성하시오. 필요한 경우 추가로 함수를 정의할 수 있다.

197 is called a circular prime because the number is a prime and its rotate values, i.e., 971 and 719 are also prime numbers. Complete the following function to test whether a given number is a circular prime. You can define extra functions if needed.

/* define extra functions here if needed */

int is_circular_prime(int n) {

}

int main() {

int num;

printf("Enter number: ");

scanf("%d", &num);

if (is_circular_prime(num))

printf("%d is a circular prime\n", num);

else

printf("%d is not circular prime\n", num);

}

2. ‘소수’는 1보다 크고 1과 자기 자신만으로 나누어지는 수이다. 예를 들어 5, 11, 23 등이 여기에 속한다. 양의 정수가 주어졌을 때 이 숫자가 소수인지 혹은 소수에 얼마나 가까운지를 알려주는 프로그램을 작성하시오.

A “prime” number is an integer greater than 1 with only two divisors: 1 and itself; examples include 5, 11 and 23. Given a positive integer, you are to print a message indicating whether the number is a prime or how close it is to a prime.

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

```
int is_prime(int num) {
```

```
}
```

```
int closest_prime(int n) {
```

```
    if (is_prime(n))
```

```
        return 0;
```

```
    else {
```

```
        int next_prime, prev_prime;
```

```
    }
```

```
}
```

```
void main(void)
```

```
{
```

```
    int n = 37;
```

```
    int p;
```

```
    printf("Enter number: ");
```

```
    scanf("%d", &n);
```

```
    p = closest_prime(n);
```

```
    if (p == 0)
```

```
        printf("Would you believe it: %d is a prime!\n", n);
```

```
    else
```

```
        printf("%d is not prime; you missed it by that much (%d)!\n\n", n, p);
```

```
}
```

다음은 위의 프로그램을 수행한 결과를 보여준다.

The following shows an execution of the above program.

```
$ ./a.out
```

```
Enter number: 37
```

```
Would you believe it: 37 is a prime!
```

```
$ ./a.out
```

```
Enter number: 33
```

```
33 is not prime; you missed it by that much (2)!
```

3. 일반적인 주사위는 1 에서 6 의 값을 출력한다. 반면에 int dice100() 은 이를 확장하여 1 에서 100 사이의 값을 출력한다. 이 주사위를 던져서 30 개의 고유한 숫자를 생성하도록 다음 프로그램을 완성하시오.

A dice returns a value between 1 and 6. On the contrary, function int dice100() randomly generates a value between 1 and 100. Complete the following program so that it would fill the array with 30 numbers unique. It is important that the numbers are not duplicated.

```
#include <stdio.h>
#include <math.h>
#include <stdlib.h>
int dice100() {
    return rand()%100 + 1;
}
int unique(int x, int output[], int n) {
    int i;

    return 1;
}
int main() {
    int i, j, p=0;
    int output[30];
    for(i = 0; i < 30; i++) {
        int x = dice100();
        while(!unique(x,output,i))
            x = dice100();
        output[p++] = x;
    }
    for(i = 0; i < 30; i++) {
        printf("%d\\n", output[i]);
    }
    return 0;
}
```

4. 다음과 같이 주어진 문자열에서 숫자만 골라 합산하는 함수 `sum_digit()`을 작성하시오. 예를 들어 다음 문자열에서 "Korea University, 2017" 숫자는 '2','0','1','7' 이다. 이들을 정수로 변환한 뒤 모두 더하면 10 이 된다. 다른 예로 "Pyeong Chang 2018 Olympic" 에서는 '2','0','1','8' 이 숫자이고 이들을 변환한 후에 모두 더하면 11 이 된다.

Write the function `sum_digit()` which converts and adds all of the numeric characters in a string. For example, "Korea University, 2017" has four numeric characters '2','0','1', and '7'. Adding them up after converting each character into the corresponding integer yields 10. As another example, "Pyeong Chang 2018 Olympic" contains the numeric characters of '2','0','1', and '8' and its sum is 11.

Hint: 문자 '0' 의 ASCII code 는 48 이다. 그러나 이 사실을 몰라도 이 C 함수를 완성할 수는 있다.
ASCII code of the character '0' is 48. However, you could write the C function without having to know this value.

```
int sum_digit(char str[]) {  
    int i;  
    int sum = 0;  
  
  
  
    return sum;  
}
```

5. “pair isogram” 이라고 불리는 단어는 단어를 구성하는 각각의 문자가 더도 덜도 아니고 정확하게 두 번씩 나타난다. 예를 들어 ‘teammate’ 와 같은 단어가 pair isogram 인데 여기서 각각의 문자가 더도 덜도 아니고 정확하게 두번씩 나타나기 때문이다. 그러나 ‘dad’의 경우는 pair isogram 이 아닌데, 그 이유는 문자 ‘a’ 가 두 번 나오지 않기 때문이다. 단어가 주어졌을 때, 그 단어가 pair isogram 인지 판별하는 프로그램을 작성하시오. 단 단어는 영문자로 구성되어 있고 모두 소문자로 가정한다.

A word is considered to be “pair isogram” if each letter in the word appears exactly twice (not less, not more) in the word. For example, the word teammate is pair isograms since each letter in the word appears exactly twice (not less, not more), but the word dad is not since the letter “a” of the word doesn’t appear twice. Given a word, you are to determine whether or not it is pair isograms. Note that all characters of the word are lower case alphabets.

```
#include <stdio.h>
int is_pair_isogram(char str[]) {
    int count[26];

    return 1;
}
void main()
{
    char str[100];
    printf("Enter word: ");
    scanf("%s", str);
    if (is_pair_isogram(str))
        printf("%s is a pair isogram\n", str);
    else
        printf("%s is NOT a pair isogram\n", str);
}
```

다음은 위의 프로그램을 수행한 결과를 보여준다.

The following shows an execution of the above program.

```
$ ./a.out
Enter word: teammate
teammate is a pair isogram
$ ./a.out
Enter word: dad
dad is NOT a pair isogram
```

6. 보안상 특별한 방에 입실할 때 id 를 입력하면 id 별로 입실 횟수를 기록하는 시스템을 개발하였다. 이 시스템은 예를 들어 최근 6 회의 입실기록을 다음과 같이 보여준다. 여기서 -1 은 입실기록의 끝을 나타낸다.

We developed a system that stores the entry history to a room per user ID for security reasons. The system prints the number of entries of each user in an increasing order in terms of user ID. The following output shows an example of the program for the recent six entries, where -1 indicates the end of history.

```
$ ./a.out
Enter id: 6927238
Enter id: 3428529
Enter id: 2800555
Enter id: 2900123
Enter id: 3428529
Enter id: 6927238
Enter id: -1
id: 2800555 (1)
id: 2900123 (1)
id: 3428529 (2)
id: 6927238 (2)
```

이 프로그램은 다음 그림 1 과 같이 linked list 구조를 이용하여 출입을 관리하고, linked list는 id 를 등록할 때 항상 정렬된 상태를 유지한다.

This program manages the data using the following linked list structure as shown in Figure 1. The order of data in the linked list remains sorted when we register an id.

```
struct Node {
    int id;
    int count;
    struct Node* next;
};

typedef struct Node* NodePtr;

void print_report() {
    for(NodePtr p = head; p != NULL; p = p->next) {
        printf("id: %d (%d)\n", p->id, p->count);
    }
}

void main() {
    int id;
    for(;;) {
        printf("Enter id: ");
        scanf("%d", &id);
        if (id < 0) {
            break;
        }
    }
}
```

```
        record_entry(id);
    }
    print_report();
}
```

예제와 같은 결과를 보이도록 다음 함수를 완성하십시오.

Complete the following function to produce the output as shown in the example.

```
void record_entry(int id) {
```

```
}
```