Assignment 4: Word Weight

4.1 Combining `fgets` and `strtok`

The tokenizing function `strtok`, can be combined with `fgets`. Once `fgets` reads a line, you may use `strtok` to separate the tokens from the line. The following shows a sample program tokenizing using `fgets` and `strtok`.

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

int main()
{
    const int MAX_LEN = 100;
    char line[MAX_LEN];

    if (fgets(line, MAX_LEN, stdin) == NULL)
        return -1;
    char *word = strtok(line, " \n");
    do {
        puts(word);
    } while (word = strtok(NULL, " \n"));

    return 0;
}
```

4.2 Programming Assignment 4: wdweight.c

Given a set of alphabetic words, print the maximum weight of the word. The weight of a word is calculated based on the weights of the characters in the word, specifically, it is the sum of the weights of characters. The weight of character c is the ordinal number of $ord(c)$ which is defined as follows:

$$ord(c) = ASCII(c) - ASCII('\texttt{A}') + 1$$

where $ASCII(c)$ is the ASCII code of c. For lower case letter c', $ord(c)$ is used for the weight of c' where c is the corresponding upper case letter for c'. As a result, the program operates in a case-insensitive manner.

For example, if you have a set of words, {`Hello`, `Coding`, `World`}, the weight of `World` is 72, which is the maximum.

The input consists of a single line in standard input. The input line consists of several words separated by space. The maximum length of the word is 50 and the number of words is 100 in maximum. The input line may contain several words of the same weight. Your program should print the maximum weight of the word.

Additional requirements for bonus points

- Do not use `scanf` nor `fscanf`, but use `fgets` or `fgetc` instead.
- Check the quality of your code to confirm that there are no style issues.

Input	Output
`Hello Coding World`	`72`
`Python C Java JavaScript Haskell`	`119`