fgets() and gets() in C language

For reading a string value with spaces, we can use either gets() or fgets() in C programming language. Here, we will see what is the difference between gets() and fgets().

fgets()

It reads a line from the specified stream and stores it into the string pointed to by str. It stops when either (n-1) characters are read, the newline character is read, or the end-of-file is reached, whichever comes first.

Syntax :

```
char *fgets(char *str, int n, FILE *stream)
str : Pointer to an array of chars where the string read is
copied.
n : Maximum number of characters to be copied into str
(including the terminating null-character).

*stream : Pointer to a FILE object that identifies an input
stream.
stdin can be used as argument to read from the standard input.


returns : the function returns str
```

- •It follow some parameter such as Maximum length, buffer, input device reference.
- •It is safe to use because it checks the array bound.
- •It keep on reading until new line character encountered or maximum limit of character array.

Example : Let’s say the maximum number of characters are 15 and input length is greater than 15 but still fgets() will read only 15 character and print it.

```c
// C program to illustrate
// fgets()
#include <stdio.h>
#define MAX 15
int main()
{
    char buf[MAX];
    fgets(buf, MAX, stdin);
    printf("string is: %s\n", buf);

    return 0;
```

```
}
```

Since fgets() reads input from user, we need to provide input during runtime.

```
Input:

Hello and welcome to GeeksforGeeks


Output:

Hello and welc
```

gets()

Reads characters from the standard input (stdin) and stores them as a C string into str until a newline character or the end-of-file is reached.

Syntax:

```
char * gets ( char * str );
str :Pointer to a block of memory (array of char)
where the string read is copied as a C string.

returns : the function returns str
```

- It is not safe to use because it does not check the array bound.
- It is used to read string from user until newline character not encountered.

Example : Suppose we have a character array of 15 characters and input is greater than 15 characters, gets() will read all these characters and store them into variable.Since, gets() do not check the maximum limit of input characters, so at any time compiler may return buffer overflow error.

```
// C program to illustrate
// gets()
#include <stdio.h>
#define MAX 15

int main()
{
    char buf[MAX];

    printf("Enter a string: ");
    gets(buf);
    printf("string is: %s\n", buf);

    return 0;
}
```

Since gets() reads input from user, we need to provide input during runtime.

```
Input:
Hello and welcome to GeeksforGeeks

Output:
Hello and welcome to GeeksforGeeks
```