* You need to strictly follow the syntaxes of that logic which you are using.

As an AI language model, I am programmed to follow the syntax rules of the logic or programming language that I am currently working with. These rules are essential to ensure that the code or logic functions as intended and to prevent errors and bugs from occurring.

Therefore, I will always try my best to follow the syntax rules of the logic or programming language that I am working with, to ensure that the code or logic I generate is valid and functional. However, if there is an error in the syntax of the input that I receive, I may not be able to generate valid output, or I may generate output that is not what the user intended.

* Write the necessary comments for better understanding to you as well as to the faculty.

As an AI language model, I don't need comments to understand code. However, comments can be helpful for human readers, including the faculty, to better understand the purpose and functionality of the code. Here are some examples of comments that could be included:

1. Comments describing the overall purpose of the code and what it is intended to accomplish.

2. Comments explaining the logic and algorithms used in the code.

3.Comments that describe the meaning and purpose of each variable and function.

4. Comments that explain any assumptions made or limitations of the code.

5. Comments that highlight any potential issues or areas where the code could be improved.

6. Comments that provide examples of input/output and expected behavior of the code.

7. Comments that describe any dependencies or requirements needed to run the code.

By including comments in the code, it can make it easier for both you and the faculty to understand the code and make any necessary modifications or improvements.

* Use this function:
* strand (time (0));

This is a line of code in the C programming language that sets the seed for the random number generator. Here's what it does:

strand () is a function in C that initializes the random number generator with a seed value.

time (0) is a function that returns the current time in seconds since the Unix epoch (January 1, 1970).

By passing the result of time (0) as the argument to strand (), we're setting the seed value to the current time, which means that the sequence of random numbers generated will be different every time the program is run.

In other words, this line of code is commonly used to initialize the random number generator with a different seed each time the program is executed, so that the sequence of random numbers generated will be unpredictable and appear more "random" to the user.

* (rand () %3) +1; -> Generating for random no.s from computer’s end.

The expression (rand () %3) +1 generates a random integer between 1 and 3, inclusive, using the C++ programming language.

Here's how it works:

The rand () function is a built-in function in C++ that generates a random integer.

The % operator computes the remainder of the division of rand () by 3, which gives a random integer in the range of 0 to 2.

Adding 1 to the result of % 3 shifts the range to 1 to 3, inclusive.

Therefore, the expression (rand () %3) +1 generates a random integer between 1 and 3, with equal probability of each value. Note that to use the rand () function, you need to include the <cstdlib> header file in your C++ program.

* Use library header files i.e., #include<stdlib.h> and #include<ctime> for above logic if necessary.

As an AI language model, I'm not sure what "above logic" you're referring to. However, I can provide an example of how to use the header files "stdlib.h" and "crime" to generate a random number between 1 and 10.

Here's an example code snippet in C++:

c

Copy code

#include <iostream>

#include <cstdlib>

#include <ctime>

using namespace std;

int main () {

srand (time (0)); // Seed the random number generator with the current time

int random number = rand () % 10 + 1; // Generate a random number between 1 and 10

cout << "Random number between 1 and 10: " << random number << endl;

return 0;

}

In this example, the "srand" function seeds the random number generator with the current time, and the "rand" function is used to generate a random number between 1 and 10. The modulo operator (%) is used to limit the range of the random number to 1 through 10, and the addition of 1 ensures that the random number is not 0. Finally, the generated random number is printed to the console using the "cout" statement.

* Add the necessary comments for better understanding of code

Sure, here's an example of code with comments added for better understanding:

python

Copy code

# This function takes in two arguments, num1 and num2, and returns their sum

def add numbers (num1, num2):

# Use the + operator to add num1 and num2

sum = num1 + num2

# Return the sum

return sum

# Call the add numbers function with the arguments 5 and 7, and print the result

result = add numbers (5, 7)

print(result)

In this code, we define a function called add numbers that takes in two arguments, num1 and num2, and returns their sum. The function first creates a variable called sum by adding num1 and num2 together using the + operator. It then returns the value of sum.

We then call the add numbers function with the arguments 5 and 7, which results in 12. We store this value in a variable called result, and then print the value of result to the console, which will output 12.