1. **DESCRIBE THE CONDITION** of a while loop. (“The condition is a...”)

2. Are the properties of the condition of a while loop the same as, or different from, the properties of the condition of an if block?

3. **WHAT ARE THE STEPS** that describe the execution of a while loop?
   
   (a)

   (b)

   (c)

4. **HOW** does a while loop **DIFFER** from an if block?
5. For each of these kinds of statements, mark **CAN** if it can appear in the body of a **while** loop, and mark **CANNOT** if it cannot appear in the body of a **while** loop. **EXPLAIN.**

(a) A named constant declaration

(b) A variable declaration

(c) A `printf` statement

(d) A `scanf` statement

(e) An assignment statement

(f) A `exit` statement

(g) An `if` block

(h) A `while` loop
6. **TRACE** the example program on slides 23 - 25 of the lecture packet titled “while Loop Lesson,” using the input values shown on slides 26 - 27. Your trace should show the following variables: `users_number`, `users_distance`, `users_last_distance` and `correct_number_hasnt BEEN input`, but in the trace you can abbreviate their names as `un`, `ud`, `uld` and `cnhbi`, respectively.
7. **DRAW A FLOWCHART** for the Infinite Loop program on slide 15 of the lecture slide packet titled “while Loop Lesson.”
8. What are the **FIVE STEPS** that describe the execution of a `for` loop?

   (a) 

   (b) 

   (c) 

   (d) 

   (e)
9. For each of these kinds of statements, mark CAN if it can appear in the body of a for loop, and mark CANNOT if it cannot appear in the body of a for loop. EXPLAIN.

(a) A named constant declaration

(b) A variable declaration

(c) A printf statement

(d) A scanf statement

(e) An assignment statement

(f) A exit statement

(g) An if block

(h) A while loop

(i) A for loop
10. Convert the following while loop into a for loop.

```c
count = initial_value;
while (count <= final_value) {
    printf("count = %d\n", count);
    count += stride;
} /* while (count <= final_value) */
```

11. Convert the following for loop into a while loop.

```c
for(count = 1; count <= n; count++) {
    n_factorial *= count;
} /* for count */
```
12. What is the **OUTPUT** of each of these programs? If you aren’t confident of an answer, type in, compile and run the program to test it.

(a) #include <stdio.h>
    int main ()
    { /* main */
      int count;
      int sum = 0;

      for (count = 1; count <= 10; count++) {
        sum = sum + count;
    } /* for count */
    printf("sum = %d\n", sum);
    return 0;
    } /* main */

(b) #include <stdio.h>
    int main ()
    { /* main */
      int count;
      int sum = 0;

      for (count = 1; count <= 10; count += 2) {
        sum = sum + count;
    } /* for count */
    printf("sum = %d\n", sum);
    return 0;
    } /* main */

(c) #include <stdio.h>
    int main ()
    { /* main */
      int count;
      int product = 1;
      for (count = 1; count <= 15; count += 5) {
        product = product * count;
    } /* for count */
    printf("product = %d\n", product);
    return 0;
    } /* main */
(d) #include <stdio.h>
int main ()
{ /* main */
    int count;
    int product = 1;

    for (count = 1; count <= 16; count += 5) {
        product = product * count;
    } /* for count */
    printf("product = %d\n", product);
    return 0;
} /* main */

(c) #include <stdio.h>
int main ()
{ /* main */
    const int lower_bound = 1;
    const int upper_bound = 17;
    const int stride = 5;
    int count;
    int product = 1;

    for (count = lower_bound;
        count <= upper_bound; count += stride) {
        product = product * count;
    } /* for count */
    printf("product = %d\n", product);
    return 0;
} /* main */
13. The combination of \( n \) items taken \( r \) at a time, denoted \( \binom{n}{r} \) and pronounced “\( n \) choose \( r \),” is the number of ways in which you can choose a subset of \( r \) items from a set of \( n \) items:

\[
\binom{n}{r} = \frac{n \cdot (n-1) \cdot (n-2) \cdot \ldots \cdot (n-r+1)}{1 \cdot 2 \cdot 3 \cdot \ldots \cdot r}
\]

For example, if you have a standard 52 card poker deck (assuming no jokers and nothing wild), the number of possible 5 card poker hands is “52 choose 5:"

\[
\binom{52}{5} = \frac{52 \cdot 51 \cdot 50 \cdot 49 \cdot 48}{1 \cdot 2 \cdot 3 \cdot 4 \cdot 5} = \frac{311,875,200}{120} = 2,598,960
\]

Write a C program that takes as its input \( n \), the total number of items available (for example, 52 cards in a deck), and \( r \), the size of the subgroup to be created (for example, 5 cards in a poker hand), and outputs \( \binom{n}{r} \). (For this homework question, you do not need to include comments if you don’t want, and you may use numeric literal constants in the body of the program.)

If you use ANY resources other than Dr. Neeman, the TAs (Glose, Ivanov, Mirza, Narasimhan), the course textbook or the materials posted on the course webpage, you MUST reference them on the quiz. THIS INCLUDES CLASSMATES, FRIENDS, PROFESSORS, ONLINE RESOURCES, ETC.