1. WHAT is a data type?

2. GIVE THREE EXAMPLES of data types.
   (a) 
   (b) 
   (c) 

3. WHAT is a variable?

4. WHAT FOUR THINGS does every variable have?
   (a) 
   (b) 
   (c) 
   (d) 

5. For each of the above four things, WHO chooses it?
   (a) 
   (b) 
   (c) 
   (d) 

6. WHICH of the above four things does the statement below cause to be set?
   int x;

7. WHICH of the above four things does the statement below cause to be set?
   float y = 22.7;
8. **WHAT** is *compile time*?

9. **WHAT** is *runtime*?

10. **WHAT** is a *declaration* (also known as a *declaration statement*)?

11. If a variable is declared but not initialized, and it has not yet been given a value, then **WHAT VALUE** does it have?

12. Some compilers on some computers automatically initialize newly declared variables to default values. **UNDER WHAT CIRCUMSTANCES** should you explicitly initialize or assign a value to a variable, rather than letting the compiler initialize it to the default value?

13. **HOW MANY VALUES** does a variable have at any given moment in runtime? **BE VERY SPECIFIC**.

14. **HOW MANY VALUES** can a variable take on over the entire duration of a run?

15. **WHAT** is the *declaration section* of a program?

16. **WHERE** in a program is the declaration section?

17. **WHAT IS THE NAME** of the other section of a program?

18. **WHERE** in a program is that other section?

19. **NAME** three ways to set the value of a variable.

   (a)

   (b)

   (c)
20. **WHAT** does an assignment statement do?

21. **GIVE AN EXAMPLE** of an assignment statement.

22. Is an assignment an **ACTION** or an **EQUATION**?

23. In an assignment, **ON WHICH SIDE OF THE SINGLE EQUALS SIGN** is the name of the variable whose value is being set?

24. In an assignment, **ON WHICH SIDE OF THE SINGLE EQUALS SIGN** is the value that the variable is being set to?

25. **WHAT** is an **initialization**?

26. **GIVE AN EXAMPLE** of an initialization statement.

27. For the initialization example above, **WHAT WOULD BE THE EQUIVALENT** if expressed as a declaration followed by an assignment?

28. In C, **WHICH CHARACTERS** can be in an **IDENTIFIER** such as a variable name?

29. In C, **WHICH CHARACTERS** can be at the **BEGINNING** of an **IDENTIFIER** such as a variable name?

30. **WHAT** is the **favorite professor rule**?

31. **MARK** valid C variable names **VALID** and invalid C variable names **INVALID**. For invalid C variable names, **EXPLAIN WHY** they are invalid. (Note that **valid** means acceptable to the compiler, rather than good programming practice.)

   (a) number_of_students_in_CS1313
   
   (b) number of students in CS1313
   
   (c) 2_to_tango
   
   (d) WHAZZAT
   
   (e) Huh?
32. For each of the following, **WRITE A DECLARATION STATEMENT** for a variable representing this quantity. For each, you should choose an appropriate data type. The name should comply with the “favorite professor” rule, and should also be a valid C identifier. **You DON’T need to initialize the variables.** Assume that int variables and float variables take 4 bytes (32 bits) each.

(a) the number of students in CS1313

(b) your height in lightyears (a lightyear is the distance that light travels in a year, which is about 6 trillion miles).

(c) a spaceship’s speed in inches per century, approximated to three significant figures (assume that the spaceship travels at 99% of the speed of light)

(d) the number of books on a bookshelf

33. **WHAT** does a placeholder **DO**?

34. **WHAT IS THE PLACEHOLDER** for each of these data types?

   (a) int
   (b) float
   (c) char

35. When a user is inputting multiple values from the keyboard, **WHICH CHARACTERS** may they use to separate the values being input?
36. **Compare** `printf` to `scanf`.

   (a) **Does it** **OUTPUT** or **INPUT**?

   `printf`:
   `scanf`:

   (b) **TO/FROM WHERE** does it output/input?

   `printf`:
   `scanf`:

   (c) **Its string literal** **CAN** or **CANNOT** contain literal text (other than a single blank space as a separator between each set of multiple placeholders)?

   `printf`:
   `scanf`:

   (d) **Its string literal** **CAN** or **CANNOT** contain a newline (for example, at the end of the string literal)?

   `printf`:
   `scanf`:

   (e) **The variable name(s) associated with placeholder(s)** **MUST** or **CANNOT** be preceded by an ampersand (`&`)?

   `printf`:
   `scanf`:
37. **WHAT IS THE OUTPUT** of each of these programs? Examine the programs **CAREFULLY**. You do not need to include extraneous blank spaces in your answer. If a program will not compile, mark **WON’T COMPILE** and **EXPLAIN**. If a program compiles and runs but does not produce any output, mark **NO OUTPUT** and **EXPLAIN**. If a program compiles and runs but produces garbage output, mark **GARBAGE** and **EXPLAIN**. If you are not confident of your answer, type in, compile and run the programs.

(a)  ```c
#include <stdio.h>
int main ()
{ /* main */
   int woopdedoo;
   woopdedoo = 127;
   printf("%d
", woopdedoo);
} /* main */
``` 

(b)  ```c
#include <stdio.h>
int main ()
{ /* main */
   int yippee = 127;
   printf("yippee = %d
", yippee);
} /* main */
``` 

(c)  ```c
#include <stdio.h>
int main ()
{ /* main */
   int oyvey = 127;
   oyvey = 128;
   printf("oyvey = %d
", oyvey);
} /* main */
``` 

(d)  ```c
#include <stdio.h>
int main ()
{ /* main */
   int ladeedah;
   printf("%d
", ladeedah);
} /* main */
```
38. **CHANGE THE ORDER** of the lines below so that the program outputs that
the mass is 5.75 kilograms. You **MUST** use **EVERY SINGLE LINE SHOWN BELOW**.

```c
/* main */
int main ()
    mass_in_kg = 10.5;
    printf("Mass: %f kg\n", mass_in_kg);
{ /* main */
    float mass_in_kg;
    #include <stdio.h>
    mass_in_kg = 5.75;
```

39. **CHANGE THE ORDER** of the lines below so that the program outputs meaningful output.
You **MUST** use **EVERY SINGLE LINE SHOWN BELOW**.

```c
{ /* main */
    printf("Let me guess your age!\n");
    scanf("%f", &age_in_years);
    printf("I’d guess that your age is %f years.\n", age_in_years);
    printf("What is your age in years?\n");
    int main ()
    #include <stdio.h>
    float age_in_years;
    age_in_years);
```