CS 330 & CS 332 Final Exam Prep

C Programming Questions – Part 1

TRUE/ FALSE

1. A preprocessor command makes your code compile faster.

1. A preprocessor command makes your code compile faster.

FALSE

1. A preprocessor command makes your code compile faster.

FALSE

Preprocessor commands have NO BEARING on compilation speed.

What do preprocessor commands do?

What do preprocessor commands do?

A preprocessor is a text substitution tool the compiler uses before performing the actual compilation. There are several commands, but the ones we’ve used most often are `#include` and `#define`.

2. A compiler converts a high-level language to executable machine code.

2. A compiler converts a high-level language to executable machine code.

TRUE

2. A compiler converts a high-level language to executable machine code.

TRUE

A compiler translates source code into machine-language instructions. Our C compilers do this by way of first converting source files into assembly, then bytecode.

3. A library is a source file that contains ready-made functions.

3. A library is a source file that contains ready-made functions.

TRUE

3. A library is a source file that contains ready-made functions.

TRUE

This is exactly what a C library is!

4. C functions cannot call themselves.

4. C functions cannot call themselves.

FALSE

4. C functions cannot call themselves.

FALSE

C functions CAN call themselves!
The language supports recursion.

5. A `struct` is a user defined data type.

5. A `struct` is a user defined data type.

TRUE

5. A `struct` is a user defined data type.

TRUE

What are the other user defined data types in C?

What are the user defined data types in C?

- `struct`
- `union`

  a collection of different data types, but only one member can contain a value.
- `typedef`

  creates an alias (new name) for a data type that already exists.
- `enum`

  consists of a set of named values.

6. Converting a variable from an `int` to a `float` will never affect its value.

6. Converting a variable from an `int` to a `float` will never affect its value.

FALSE

6. Converting a variable from an `int` to a `float` will never affect its value.

FALSE

While both `int`s and `float`s are 4 bytes in size, a large enough `int` would get truncated when converted to a `float`, because not all its width is used to represent a whole number.

7. `while` loops are faster than `for` loops.

7. `while` loops are faster than `for` loops.

FALSE

7. `while` loops are faster than `for` loops.

FALSE

`while` loops are NOT faster than `for` loops.

8. A `do-while` loop will always execute at least once.

8. A `do-while` loop will always execute at least once.

TRUE

8. A `do-while` loop will always execute at least once.

TRUE

A `do-while` loop will execute its statement first, BEFORE checking the loop condition.

9. C is an object-oriented language.

9. C is an object-oriented language.

FALSE

9. C is an object-oriented language.

FALSE

While `struct`s allow us to implement some OOP principles in the language, C is NOT object-oriented.

10. C is a low-level language.

10. C is a low-level language.

FALSE

10. C is a low-level language.

FALSE

While C gets closer to the wire than the languages we may have learned prior, (Python, Java), C is itself a high-level language.

11. The **&** and **&&** operators are functionally equivalent

11. The **&** and **&&** operators are functionally equivalent

FALSE

11. The **&** and **&&** operators are functionally equivalent

FALSE

& is the bitwise and operator and **&&** is the logical and operator.

12. The size of a pointer is always 8 bytes.

12. The size of a pointer is always 8 bytes.

TRUE

12. The size of a pointer is always 8 bytes.

TRUE

Regardless of the data type to which it is pointing, a pointer is always 8 bytes wide.

MULTIPLE CHOICE

13. Which standard library includes the `printf()` and `scanf()` functions?

A. `<time.h>`
B. `<stdlib.h>`
C. `<stdio.h>`
D. `<printer.h>`

13. Which standard library includes the `printf()` and `scanf()` functions?

A. `<time.h>`
B. `<stdlib.h>`
C. `<stdio.h>` CORRECT
D. `<printer.h>`

14. What will the following program print?

```
main {
    float f = 9.45;
    int i = f;
    i += 0.55;
    f = i;
    printf("%f", f);
}
```

A. 9.0
B. 10.0
C. 9.55
D. 1.0

14. What will the following program print?

```
main {
    float f = 9.45;
    int i = f;
    i += 0.55;
    f = i;
    printf("%f", f);
}
```

A. 9.0 CORRECT
B. 10.0
C. 9.55
D. 1.0

15. What is the correct format specifier to print characters?

A. `%d`
B. `\c`
C. `%c`
D. `%lf`

15. What is the correct format specifier to print characters?

A. `%d`
B. `\c`
C. `%c` CORRECT
D. `%lf`

16. What will the following program print?

```
main {
   int i;
   for (i = 0; i < 10; i++)
      i += 2;
   printf("%d", i);
}
```

A. 9
B. 10
C. 11
D. 12

16. What will the following program print?

```
main {
    int i;
    for (i = 0; i < 10; i++)
        i += 2;
    printf("%d", i);
}
```

A. 9
B. 10
C. 11
D. 12
CORRECT

17. Between `gets()` and `fgets()`, which is the safer function?

A. `gets()`
B. `fgets()`
C. They are equally safe
D. They are equally unsafe

17. Between `gets()` and `fgets()`, which is the safer function?

A. `gets()`
B. `fgets()` CORRECT
C. They are equally safe
D. They are equally unsafe

18. What will the following program print?

```
main {
   int a, b;
   a = b = 50;
   b /= 2;
   a *= 2;
   printf("%d", ++a + b--);
}
```

A. 126
B. 125
C. 100
D. error

18. What will the following program print?

```
main {
   int a, b;
   a = b = 50;
   b /= 2;
   a *= 2;
   printf("%d", ++a + b--);
}
```

CORRECT

A. 126

B. 125

C. 100

D. error

19. Which method is used to convert an integer to a char/string data type?

A. `atoi()`
B. `itoa()`
C. `itos()`
D. `ctoi()`

19. Which method is used to convert an integer to a char/string data type?

A. `atoi()`
B. `itoa()` CORRECT
C. `itos()`
D. `ctoi()`

20. What will the following program print?

```
main {
    printf("%d", ((3/4) * 60) + 14);
}
```

A. 59
B. 0
C. 14
D. 45

20. What will the following program print?

```
main {
    printf("%d", ((3/4) * 60) + 14);
}
```

A. 59
B. 0
C. 14 CORRECT
D. 45

21. Below is a list of different variables. Which option lists these variables by descending size?*

```
char s[5];
int i;
long l;
struct mystruct {
    char x;
    char y;
    int z;
} STRUCT;
char c = '\n';
```

A. s, STRUCT, i, l, c

B. l, s, STRUCT, i, c

C. STRUCT, l, s, i, c

D. l, STRUCT, s, i, c

*Assume that the compiler is NOT adding padding to align data.

21. Below is a list of different variables. Which option lists these variables by descending size?*

```
char s[5];
int i;
long l;
struct mystruct {
    char x;
    char y;
    int z;
} STRUCT;
char c = '\n';
```

A. s, STRUCT, i, l, c

B. l, s, STRUCT, i, c

C. STRUCT, l, s, i, c

CORRECT

D. l, STRUCT, s, i, c

*Assume that the compiler is NOT adding padding to align data.

22. If we have some variable `int var`, `&var` would give us:

A. The address of `var`.
B. The data type of `var`.
C. The size of `var` (in bytes).
D. The value at the location of `var`.

22. If we have some variable **`int var`**, **`&var`** would give us:

A. The address of **`var`**. CORRECT
B. The data type of **`var`**.
C. The size of **`var`** (in bytes).
D. The value at the location of **`var`**.

23. If we have some variable `int var`, `*var` would give us:

A. The address of `var`.
B. The data type of `var`.
C. The size of `var` (in bytes).
D. The value at the location of `var`.

23. If we have some variable `int var`, `*var` would give us:

A. The address of `var`.
B. The data type of `var`.
C. The size of `var` (in bytes).
D. The value at the location of `var`. CORRECT

24. Given the following code, what would the output of `ptr2` be?

```
int a = 5;
int *ptr1 = &a;
int **ptr2 = &ptr1;
```

A. 5
B. The address of `ptr1`
C. The address of `a`
D. The value of `ptr1`

24. Given the following code, what would the output of **`ptr2`** be?

```
int a = 5;
int *ptr1 = &a;
int **ptr2 = &ptr1;
```

A. 5
B. The address of **`ptr1`** CORRECT
C. The address of **`a`**
D. The value of **`ptr1`**

Thank you for coming!

Please write your blazerid on the whiteboard on your way out.