COL7001 Practice Midterm

1. Define the role of each of the following:
   a) Compiler
   b) Assembler
   c) Loader
   d) Linker
2. Distinguish between **compile-time errors** and **runtime errors** with one example each.
3. What is the role of a **bootloader** during system startup?
4. What is one key difference between an **interpreted language** (like Python) and a **compiled language** (like C)?
5. Suppose a program contains the following global and local variables:

```
int g = 10;   // global
int main() {
        int x = 5;
        char *p = malloc(100);
}
```

   Identify where each variable (g, x, and p) is allocated in memory (code, stack, heap, or data segment).

6. Compare and contrast **static linking** and **dynamic linking**. Give an example scenario where dynamic linking is preferable.
7. Explain the role of the **system call interface** in an operating system. Why can't user programs directly execute privileged operations such as I/O?
8. Draw a typical process address space layout showing segments that we referred to in class.
9. If the program makes a recursive call that exceeds stack size, what error occurs? Explain.
10. Briefly outline the steps needed to **add a new system call in Linux** (kernel modification, syscall table update, recompilation).
11. Define the following terms in one or two sentences each:
    - Pipelining
    - Out-of-order execution
    - Superscalar architecture
12. Explain how instruction pipelining improves CPU performance. Illustrate with a 5-stage pipeline example (IF, ID, EX, MEM, WB).
13. Suppose you have a 4-way superscalar processor. If the compiler cannot extract much instruction-level parallelism (ILP), will the processor still achieve 4× speedup? Why or why not?
14. A program frequently accesses the following:

    - Variable X stored in L1 cache
    - Variable Y stored in DRAM
    - Large dataset stored on disk

    Rank the access times (approximate orders of magnitude: nanoseconds, microseconds, milliseconds) and discuss how the **principle of locality** helps optimize performance.

15. Questions that I asked in class for you to go back and look up e.g, printf code and its relation to libc,