These problems were generated by TAs and instructors in previous semesters. They may or may not match the actual difficulty of problems on Test5.

Levels of Concurrency

1. Answer the following True/False questions and explain the reasoning behind your answer.
    a. In multitasking, concurrent programs are run on a single CPU

    b. Multitasking and multiprocessing are synonyms meaning the same thing.

    c. A CPU has a single logic unit for each basic arithmetic operation (ex: multiplication, addition)

    d. Concurrency means that programs are running at the exact same time.

2. If the following algorithm were made concurrent, what is the minimum number of time steps it could take? List the steps (which operation(s) would be done in each time step) and explain. **2*(6+7)-9**

3. What is the last level of concurrency? Briefly explain what this level of concurrency is and what are some applications we can see in daily life.

4. Which of the following best describes the features of a distributed system? **Select all that apply.**

    ❏ Has many connected computers.
    ❏ Has only one computer with many cores.
    ❏ Can perform multitasking.
    ❏ Can perform multiprocessing.
    ❏ Can perform pipelining of tasks.
    ❏ None of the above.

Parallel Programming

1. What is the definition of deadlock? How can you usually fix deadlock?

2. A factory that makes action figures is set up so that every worker does three tasks: collecting the components (5min), gluing them together (5min), and painting the resulting figure (5min). Gluing requires some tidy-up time before switching to a new task (5min), and painting requires both set-up and tidy-up time when switching to a new task (5min each), so the worker's schedule over an hour currently looks like this:

	00:00	00:05	00:10	00:15	00:20	00:25	00:30	00:35	00:40	00:45	00:50	00:55
X	C	G	T	S	P	T	C	G	T	S	P	T
Y	C	G	T	S	P	T	C	G	T	S	P	T
Z	C	G	T	S	P	T	C	G	T	S	P	T

X, Y, and Z are workers. C is collect, G is glue, P is paint, T is tidy, and S is set-up

How many fully-set-up figures can 3 workers currently produce in 1 hour?

Use the concept of **pipelining** to adjust this schedule so that the workers can generate more figures in an hour. Fill in the table below using the same codes we used above. Your new schedule does not need to be ideal; it should just be better than the old one.

	00:00	00:05	00:10	00:15	00:20	00:25	00:30	00:35	00:40	00:45	00:50	00:55
X												
Y												
Z												

How many fully-set-up figures can 3 workers produce with your new schedule in 1 hour?

Internet

1. What does it mean to be fault tolerant? At a high level (no specific examples needed) how is the Internet structured to allow it to be fault tolerant?

2. What is a packet? Name one thing that could go wrong when sending a packet, and how the Internet deals with it.

3. What is the relationship between an IP Address and a URL?

4. For each of the following statements, select whether it is True or False.
    a. Websites split up data into a few large packets based on content before sending them through the internet.

    b. Packets all arrive at your browser in the correct order, to support buffering.

    c. Packets might all take different paths through routers to reach your computer.

Encryption

1. Explain why RSA is secure. What might change that might make RSA NOT secure?

2. Explain how asymmetric keys can be more secure than symmetric keys

3. What is the difference between data privacy and data security? What do the two have in common?

4. How can encryption help to stop Man-in-the-Middle Attack?

5. Given the following scenarios, determine what type of security attack it is: DDOS or Man in the Middle.
    a. An attacker sets up a router and allows other people to connect to it. The attacker reads every packet that people send and records usernames and passwords.

    b. An attacker wants to prevent people from receiving website content, so they send many packets to the website's server to overwhelm it.

6. Given the code below, what string will the variable `msg` hold after the code runs?

```
def encrypt(s, shift):
    result = ""
    for c in s:
        if c != " ":
            result = result + chr(ord(c) + shift)
        else:
            result = result + c
    return result

s = "Hello"
msg = encrypt(s, 3)
```

Next, write a single line of code that will correctly decrypt the message in the variable `msg` and print the result. You may not use the variable `s` in this line of code or hardcode the original message in any way.

Hint: you should call the function `encrypt` again, but what should `shift` be?