1) Rewrite
\[ F(a, b, c, d) = a'b'c'd' + a'b'c d' + a'b'c d + a'bc'd' + a'bc d' + ab'c'd' + ab'c d' \]
in fully simplified sum-of-products form.

2) Consider the following two attempted solutions to the 2-process mutual exclusion problem. For each attempt, answer yes/no with a brief justification.
   a) Does the code guarantee mutual exclusion?
   b) Is it possible that both processes will busy-wait forever? That is, could deadlock occur?
   c) Does the code guarantee fairness? That is, is indefinite postponement impossible? Briefly explain your answers.

**Attempt #1**: common variables: flag1, flag2 (both initially false)

```c
Process 1
while (true) {
    flag1 = true;
    while (flag2); //empty body
    Critical section;
    flag1 = false;
    Noncritical section;
}
```

```c
Process 2
while (true) {
    while (flag1); //empty body
    flag2 = true;
    Critical section;
    flag2 = false;
    Noncritical section;
}
```

**Attempt #2**: common variable: lock (initially false)
Assume the existence of an atomic (non-interruptible) test_and_set function that both returns the value of its boolean argument and sets the argument to true.

```c
Process 1
while (true) {
    while (test_and_set(lock));
    Critical section;
    lock = false;
    Noncritical section;
}
```

```c
Process 2
while (true) {
    while (test_and_set(lock));
    Critical section;
    lock = false;
    Noncritical section;
}
```
3) Consider a system with 3 resources (A, B, C) in quantity (7, 7, 6). The Banker’s Algorithm is used to allocate resources and it has the following SAFE state:

Available:  A  B  C
            1  2  2

<table>
<thead>
<tr>
<th>Process</th>
<th>Allocation</th>
<th>Max</th>
<th>Need</th>
</tr>
</thead>
<tbody>
<tr>
<td>P0</td>
<td>2 1 1</td>
<td>2 4 4</td>
<td>0 3 3</td>
</tr>
<tr>
<td>P1</td>
<td>1 1 2</td>
<td>2 4 4</td>
<td>1 3 2</td>
</tr>
<tr>
<td>P2</td>
<td>3 2 1</td>
<td>6 6 1</td>
<td>3 4 0</td>
</tr>
<tr>
<td>P3</td>
<td>0 1 0</td>
<td>0 3 2</td>
<td>0 2 2</td>
</tr>
</tbody>
</table>

a) Justify why the current state is safe.
b) For each part, write your choices on your solution sheet. You do not need to justify your answers.
i) Select a process and a request of a single instance of an available resource where the request will be denied. The resource must be within the specified need for that process.

Process ______ Resource ______

ii) Select a process and a request of a single instance of an available resource where the request will be allowed.

Process ______ Resource ______

Choose any 2 problems.

1) Write the function

```cpp
insert_double (*NodeType head, int key)
```
to insert a new integer key into a sorted non-empty doubly linked list beginning at address head. Declare all data structures.

2) Write the function

```cpp
int count2children(treeptr p);
```
that is given a (possibly empty) binary tree and returns the number of nodes in the tree that have both a left child and a right child.

3) Solve the recurrence relation $T(n) = 2T(n/2) + 5$ where $T(1) = 1$ and $n = 2^k$ for a nonnegative integer $k$. Your answer should be a precise function of $n$ in closed form. (An asymptotic answer is not acceptable.) Justify your solution.
Answer ANY TWO of the following three questions:

1. Give the state diagram for a deterministic finite automaton (DFA) that recognizes the following regular language over $\Sigma = \{a, b, c\}$:
   
   $$\{w : w \text{ contains at least one } a, \text{ one } b, \text{ and one } c \text{ in any order}\}$$

2. Prove that the following language over $\Sigma = \{a, b, c\}$ is not context-free:
   
   $$\{w : w \text{ contains the same number of } a' \text{'s and } b' \text{'s and } c' \text{'s in any order}\}$$

3. Let $A_{TM} = \{M, w : M \text{ is a Turing machine that accepts string } w\}$
   
   Let $TWO_{TM} = \{M : M \text{ is a Turing machine that accepts exactly two strings}\}$

   Show that $A_{TM} \leq TWO_{TM}$. 