1.
    a) [5 points] Construct a DFA $M_1$ for which $L(M_1)$ is the set of strings over 0, 1 with an odd number of 0s.
    b) [5 points] Construct a DFA $M_2$ for which $L(M_2)$ is the set of strings over 0, 1 for which 101 does not appear as a substring.
    c) [10 points] Construct a DFA $M_3$ for which $L(M_3) = L(M_1) \cap L(M_2)$. Explain the method you used to construct $M_3$.

2.
    a) [10 points] Construct a DFA $M$ for which $L(M)$ is the set of strings over 0, 1 where both the number of 0s and the number of 1s are evenly divisible by 1000. **Hint:** You will need to describe this DFA parametrically. For example, if you had states labeled in terms of integers $i$ and $j$, you could describe $\delta(q_{i,j},...)$ in terms of $i$ and $j$.
    b) [15 points] Formally prove that your construction from 2a) is correct.

3. [20 points] Convert the following NFA to a DFA, and informally describe the language it accepts. **Hint:** It is okay to start by “adjusting” the original NFA by combining or removing states if you want. If you do so, however, clearly explain the changes you made and why your changes don’t affect the language accepted by the NFA. (I will accept informal explanations for this.) Drawing transition diagrams may be helpful.

```
        0  1
→ p   {p, q}  {p}
 q    {r, s}  {t}
 r    {p, r}  {t}
 *s   ∅  ∅
 *t   ∅  ∅
```

4. Consider the following NFA with $\epsilon$-transitions:

```
        ε   a   b   c
→ p   {q, r}  ∅  {q}  {r}
 q    ∅  {p}  {r}  {p, q}
 *r   ∅  ∅  ∅  ∅
```

    a) [5 points] List the $\epsilon$-closure of each state.
    b) [10 points] Convert this automaton to an NFA without $\epsilon$-transitions.

5.
    a) [10 points] Write a regular expression over $\Sigma = \{0, 1\}$ that accepts the set of all strings where every occurrence of 1 is preceded by at least three 0s.
    b) [10 points] Using any strategy you want, create a DFA that accepts the same language over $\Sigma = \{0, 1\}$ as $00^* + 11^*$. 