1. Let $L$ be a language over $\{0, 1\}$ that consists of strings having even number of zeroes or odd number of ones. For example, $1010, 1101 \in L$ and $101 \notin L$. ($5 \times 2$)

(i) Design a DFA for $L$. (No correctness proof required)

(ii) Hence or otherwise write a regular expression for this language. (No correctness proof required)

Even no. of 0's $r_1 = 1^* \cdot (01^*0)^*1^*$
Odd no. of 1's $r_2 = 0^* \cdot 10^*1^*0^*$
Ans is $r_1 + r_2$.

Comment: Although proof was not required, some basis must have been given for the regular expression. Simply writing a long regular expression without saying what it represents was not acceptable. I also found several answers trying to capture all paths which was not done in class (it comes from writing and solving state equations). Such answers were also penalized.
2. Are the following languages regular? Justify. (5 × 2)

(a) \( L = \bigcap_{i=1}^{\infty} L_i \) where \( L_i \) is regular.

Not regular. We have already seen in class that \( L = \bigcup_{i=1}^{\infty} L_i \) is not regular. Since regular languages are closed under complementation, \( L = \bigcup_{i=1}^{\infty} L_i = \bigcap \bar{L}_i \).

If \( \bigcap_{i=1}^{\infty} L_i \) is regular, then \( \bigcap_{i=1}^{\infty} \bar{L}_i \) is regular and consequently its negation is regular which will contradict that \( L = \bigcup_{i=1}^{\infty} L_i \) may not be regular.

**Comment** By rewriting \( L = \bigcap_{i=1}^{\infty} L_i \) as \( \bar{L} = \bigcup_{i=1}^{\infty} \bar{L}_i \) and then invoking the previous result on infinite union is not the same unless done with some extra care. You have to first show that \( L = \bigcup_{i=1}^{\infty} L_i \) is not regular, implies that \( L = \bigcup_{i=1}^{\infty} \bar{L}_i \) is not regular and so on.

We are using the mechanism of reduction to show that the current statement (say \( A \)) would imply that a known statement (\( B \)) is true. Since \( B \) is not true so \( A \) cannot be true.

The statement \( B \) proved in class had the form \( L = \bigcup_{i=1}^{\infty} L_i \) so any other form (including negation) must be explicitly dealt with.

(b) \( L = \{a_1a_2\ldots a_k|a_i \in \Sigma \} \) where \( a_1 \cdot a_3 \cdot a_5 \ldots a_{k-1} \in L_1 \) \( a_2 \cdot a_4 \ldots a_k \in L_2 \) and \( L_1, L_2 \) are regular.

\( L \) is regular. We can design a DFA similar to the product construction, with an extra bit in the state space to alternate between \( M_1 \) and \( M_2 \) (corresponding to \( L_1 \) and \( L_2 \)). So \( Q = Q_1 \times Q_2 \times \{1, 2\} \) and \( \delta((q_1, q_2, 1), a) = (\delta_1(q_1, a), q_2, 2) \) and similarly for \( (q_1, q_2, 2) \). For acceptance, it must be \( F_1 \times F_2 \times 1 \) (must end on the final state of \( M_2 \)).

**Comment** If you only used some intuitive constructions with block diagrams and non-det transitions without giving the formal state-space and transition function, it was not acceptable.
3. Let \( L_{\text{pal}} = \{ w \in (0+1)^* | w \text{ is a palindrome} \} \). A palindrome is a string which equals its reversal. For example, 01010 is a palindrome but 01011 is not a palindrome. Use pumping lemma to show that \( L_{\text{pal}} \) is not regular. (8)

Let \( L_{\text{pal}} \) be regular and \( n \) be the constant of the pumping lemma for \( L_{\text{pal}} \). Choose the following string \( z = 0^n \cdot 1^n \cdot 1^n \cdot 0^n \). Then \( u \cdot v \cdot w = z \) where \( |v| = 0^k \) for some \( 1 \leq k \leq n \). and \( uv^i w \in L_{\text{pal}} \) for all \( i \). This is not possible since the string will have unequal number of 0’s in the beginning and end which cannot form a palindrome.

4. Let \( L = \{ 0^k | k \text{ is prime or even} \} \). Is \( L \) regular? Justify (12)

The given language is not regular. Let us assume that 2 is not prime to keep the argument simpler. Let \( L_1 = \{ 0^k | k \text{ is even} \} \) and \( L_2 = \{ 0^k | k \text{ is prime} \} \). So \( L = L_1 \cup L_2 \).

Let us assume that \( L \) is regular. Then \( L - L_1 = L \cap \overline{L_1} = L_2 \). We know that \( L_2 \) is not regular (primes done in class). So from closure properties we arrive at a contradiction about \( L_2 \). Therefore \( L \) cannot be regular.

By including 2 in \( L_2 \), we need an extension of the result on primes, namely, \( L_3 = \{ 0^k | k \text{ is prime greater than 2} \} \) is also not regular. This also follows from closure property since \( \{ 2 \} \) is regular, \( L_3 \cup \{ 2 \} \) will be regular - contradiction.

Alternate approach using PL only - no closure properties are used

Choose a string \( 0^n \) where \( n > 2 \) is an odd prime. Then \( u \cdot v \cdot w = 0^n \) where \( |u| = x \) \( |v| = y \geq 1 \) \( w = n - x - y \). Then for any \( i \geq 0 \), \( 0^n \) is in \( L \) where \( n_i = x + iy + n - x - y = iy - n - y = y(i - 1) + n \).

Choose \( i = 2n + 1 \). So \( n_i = (n)(2y + 1) \). Then \( n_i \) is not prime and it is also not even since it is a product of two odd numbers. This leads to a contradiction.