Ans 1)

It’s the set consisting of 0s and 1 where the number of ones is equal to the number of blocks of zeroes. Ones are never consecutive. All strings end with 1 and begin with 0. There can be odd as well as even blocks of zeroes. The relationship between the (2n-1)th and the (2n)th block of zeros is that the latter has twice the number of zeros as the prior block. The relationship between the (2n)th and the (2n+1)th block is that the latter has one more zero than the prior numbered block. The shortest string is 01.

A few strings

{01, 0101, 010^210^310^61, 010^210^310^610^710^{14}1, ...}

The last string -- 010^210^310^610^710^{14}1 is the string with 6 blocks.

Ans 2)

\begin{align*}
\text{a)} & \quad h(0120) & = & \text{aabbba} \\
\text{b)} & \quad h(21120) & = & \text{baababbaa} \\
\text{c)} & \quad h(L) & = & \text{a(ab)*ba} \\
\text{d)} & \quad h(L) & = & \text{a+abba} \\
\text{e)} & \quad \text{Inv}_h(ababa) & = & \{022,110,102\} \\
\text{f)} & \quad \text{Inv}_h(L) & = & L(02^*) \cup L(1^*02^*)
\end{align*}

Ans 3)

Consider any language L over alphabets \{a,b\}
Define homomorphism h1 as below
\begin{align*}
h1(a) & = \text{a} \\
h1(a') & = \text{a} \\
h1(b) & = \text{b}
\end{align*}
consider the regular language L1 defined by the regular expression (a+b)*a’
Define homomorphism h2 as below
\begin{align*}
h2(a) & = \text{a} \\
h2(a') & = \text{c} \\
h2(b) & = \text{b}
\end{align*}
□ is for intersection

Now we can apply \( h2(\text{Inv}_h1(L) \sqcap L1) \) will give language \( L/a \)
Since regular languages are closed over inverse homomorphism, homomorphism
and intersection and we began with a regular language \( L \) the result, \( L/a \), is also
regular.

Ans 4)

(a) \( \text{min}(L) \) = \{ w | w is in L, but no proper prefix of w is in L \}

Construct a DFA, \( D1 \), for \( L \) (since it is regular we will always be able to
do so). Remove all outgoing transitions from the final states of \( D1 \).
The resulting construction accepts the language \( \text{min}(L) \) and is therefore
regular.

(b) \( \text{max}(L) \) = \{ w | w is in L and for no x other than \( \epsilon \) is wx in L \}

Construct a DFA, \( D1 \), for \( L \) (since it is regular we will always be able to
do so). For all final states of \( D1 \) check if there is a path from that final
state to another final state (include it self) if so mark all such states.
Remove all the marked states from the set of final states. The resulting
DFA accepts the language \( \text{max}(L) \) and is therefore regular.

(c) \( \text{init}(L) \) = \{ w | for some x, wx is in L \}

Construct a DFA, \( D1 \), for \( L \) (since it is regular we will always be able to
do so). For all states (assuming \( x = \epsilon \) is not allowed) check if there is a
path from that state to a final state, if so mark the state. Make the marked
states as the final states of the new DFA. The language accepted by the
resulting DFA is \( \text{init}(L) \) and so its regular.
Ans 5)

L and M are the languages and we need to prove alt(L,M) is regular given L and M are regular, where alt(L,M) is defined as in the book.

We run the machines accepting L and M in parallel and have a mod 2 counter logic (that accepts strings of even length including 0) that feeds the input to only one machine at a time (starting first with L). The if we have a string which for which both the machines for L and M are in final states and the mod 2 counter is in the start state (ie state for 2,4,6) we accept.

The set of states are the triplets [l,m,c]

Where  l is from the set of states of automata accepting L
m is from the set of states of automata accepting M
c is from the set of states of C

The set of final states are the ones where all three -- l, m and c are final states in their respective automatons

The accepting state for C is c0

Assuming without loss of generality that the language contains 0s and 1s, the transition functions will be

\[ q([l1,m1,c0], a) = [q(l1,a),m1,c1] \text{ where } a \text{ is } (0+1) \]
\[ q([l1,m1,c1], a) = [l1,q(m1,a),c0] \text{ where } a \text{ is } (0+1) \]