Multiple Choice:

|  |  |
| --- | --- |
| 1 | bc |
| 2 | ac |
| 3 | cd |
| 4 | bd |
| 5 | bd |
| 6 | bd |
| 7 | ad |
| 8 | bc |
| 9 | ac |
| 10 | bc |
| 11 | ad |
| 12 | ac |
| 13 | bc |
| 14 | bd |
| 15 | cd |
| 16 | bc |
| 17 | bd |
| 18 | ac |
| 19 | bc |
| 20 | bc |
| 21 | ac |
| 22 | ad |
| 23 | bc |
| 24 | ac |
| 25 | ac |
| 26 | bd |
| 27 | cd |

Long Answers:

1a.

A is Halt\_TM because, by letting B be NE\_TM and if there exists a decider for NE\_TM, we can use it to decide Halt\_TM (see below).

1b.

B is NE\_TM because, by letting A be Halt\_TM and if there exists a decider for NE\_TM, we can use it to decide Halt\_TM (see below).

1c.

Suppose x = <M, w> is in A (Halt\_TM), meaning that the Turing Machine M halts on w. Since M halts, M either accepts or rejects w:

* If M accepted w, then we could let x = 101, and M’ would’ve accepted x, meaning that the language of M’ contains x and is therefore not empty.
* If M rejected w, then M’ would accept any string x, so the language of M’ is not empty.

Therefore, since the language of M’ is always non-empty, M’ is in B (NE\_TM).

1d.

Suppose x = <M, w> is not in A (Halt\_TM), meaning that the Turing Machine M loops on w. In this case, the machine M’ would wait forever to know which of the if-else-branches to go into. Therefore, the language of M’ is empty because we cannot find a single string x that M’ ever accepts, implying that M’ is not in B (NE\_TM).

2.

3a.

a=true, b=true, c=false.

3b.

3c.

3d.