1. Write the microinstructions required to do the following, similar to slides 52-57 in lecture 3 (optimize as much as you can):
   a. Fetch
      \[6\text{ points } \rightarrow 1\text{ for each micro-inst and 1 point for optimization}\]
      \begin{align*}
      \text{PC} & \rightarrow \text{MAR} \\
      \text{M}[\text{MAR}] & \rightarrow \text{MDR} \\
      \text{AIR+4} & \rightarrow \text{PC} \\
      \text{MDR} & \rightarrow \text{IR} \\
      \end{align*}
      then the decoding
   
   b. Addi R1, R2, 5 (i.e. R1 = R2 + 5)
      \[2\text{ points}\]
      \begin{align*}
      \text{R}[2] & \rightarrow \text{AIR} \\
      \text{AIR} + \text{SE(imm)} & \rightarrow \text{R}[1] \\
      \end{align*}
   
   c. lw R5, offset(R6) (i.e. R5 = Memory[R6 + offset])
      \[4\text{ points}\]
      \begin{align*}
      \text{R}[6] & \rightarrow \text{AIR} \\
      \text{SE(IR<offset>) + AIR} & \rightarrow \text{MAR} \\
      \text{M}[\text{MAR}] & \rightarrow \text{MDR} \\
      \text{MDR} & \rightarrow \text{R}[5] \\
      \end{align*}

2. \[3\text{ points}\] We will not benefit anything. Actually we lose. With a second AIR, we will need to fill both AIR and AIR2 and then start the operation at ALU. So, we paid for extra hardware and didn’t gain performance.

3. \[2\text{ points}\] If we remove MAR, the design won’t work. What if PC is sent to the memory but this memory is still busy serving a previous operation? This is why MAR is here to buffer the address till the memory is ready.

4. \[3\text{ points}\] No, it won’t be beneficial. The memory is much slower than the datapath. If we connect it to the bus, we are forcing all other parts connected to the bus to work at the slow speed of the memory, hence losing performance.