T1

The purpose of the .END pseudo-op is to tell the assembler where the assembly language program ends. The differences between .END and HALT is that, HALT will stop the program, but the .END will not do that.

T2

Queue is a kind of data structure that keeps the element who comes in first will also go out first (FIFO).

T3

Those two .FILL instructions should be put after the HALT instruction.

```
.ORIG x3000
LD R0, A
ST R0, B
HALT
A    .FILL xDEAD
B    .FILL xBEEF
.END
```

T4

Yes, there would be a problem if you use the label 'AGAIN' in both modules without declaring it as external.

In assembly language, a label is a symbolic name for a certain location in the code[1]. If two separate modules both use the label 'AGAIN', the linker would not know which 'AGAIN' to refer to when linking the modules together. This could lead to unexpected behavior or errors.

The '.EXTERNAL' directive is used to tell the assembler that a certain label is defined in another file. If neither module contains the pseudo-op '.EXTERNAL AGAIN', the assembler has no way of knowing that 'AGAIN' is used in more than one module.

T5

1. x0FFF
2. Yes, it will.
3. The program will stuck in the last instruction wihout stopping.

T6

.FILL Occupies an address and fills the initial value with the memory cell that the address points to.

.BLKW Occupies contiguous address space.

.STRINGZ The address space is occupied consecutively and initialized, and the last unit of memory is set to x0000.

T7

```
        .ORIG x3000
        LD R0, A
        LD R1, B
X       NOT R2, R0
        ADD R2, R2, #1
        ADD R2, R2, R1
        BRz DONE
        ADD R1, R1, #-1
        ADD R0, R0, #1
        BRnzp X
DONE ST R1, C
        TRAP x25
A       .BLKW 1
B       .BLKW 1
C       .BLKW 1
```

T8

```
      .ORIG x3000
      ; Suppose R0 is already loaded with the target number
      ; Initialize
      AND R1, R1, #0 ; Result
      ADD R2, R1, #15 ; Loop var i
      ADD R3, R1, #2 ; Mask
      ADD R4, R1, #1 ; 1 << (15 - i)
      AND R5, R5, #0 ; Temp result
      ; Main Loop
L     AND R5, R3, R0 ; Test bit
      BRz N ; Go check next bit
      ADD R1, R1, R4 ; Add to result
N     ADD R3, R3, R3 ; Check bit from bit 2
      ADD R4, R4, R4 ; L-shift R4
      ADD R2, R2, #-1 ; Control the loop
      BRp L
      ; End
      HALT
      .END
```

T9

1. A, H.
2. After PUSH F or after PUSH G.
3. It contains nothing.

T10

```
        .ORIG x3000
        LD R1, VALUE
        NOT R1, R1
        ADD R1, R1, #1
        ADD R1, R1, R6
        BRz EMPTY
        LDR R0, R6, #0
        BRnzp OVER

EMPTY   LEA R0, A
        TRAP x22
OVER    HALT

VALUE   .FILL x4000
A       .STRINGZ "UNDERFLOW"

        .END
```