Question 1 (30 points): This is a multiple-choice question. Unless otherwise stated, there is only one correct answer for each question.

a. The frame pointer is used in MIPS to ensure that the base address, for all references to the stack frame of a function, does not change during the execution of the function. Thus the use of the frame pointer is highly recommended in a function that:

(a) ( ) is a leaf function that does not call another function.
(b) ( ) is a recursive function.
(c) (X) makes multiple allocations in the stack.
(d) ( ) uses the stack to save callee-saved registers.
(e) ( ) has lots of parameters and some of them have to be passed in the stack.

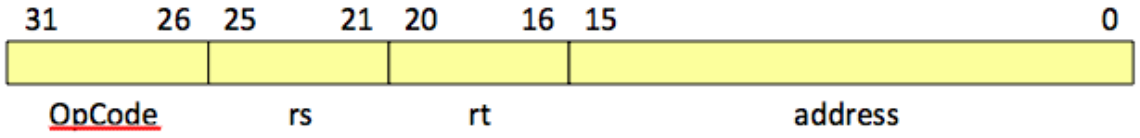
b. You have to write MIPS assembly code to execute the following C language statement:

```
A[B[j]] = x+y
```

The values of `A`, `B`, `j`, `x`, and `y` are in memory. The minimum number of loads and stores that the MIPS code that executes this statement must have is:

(a) ( ) 5 loads and 1 store;
(b) (X) 6 loads and 1 store;
(c) ( ) 2 loads and 1 store;
(d) ( ) 2 loads and 2 stores;
(e) ( ) 1 loads and 1 store;

c. The format and the effect of a branch instruction are shown below:



$$\text{beq rs, rt, address} \Leftrightarrow \text{PC} \leftarrow \text{PC} + 4$$
$$\text{if(rs} = \text{rt)PC} \leftarrow \text{PC} + \text{sign\_extended(address} << 2)$$

The *target* of a branch is the next instruction executed when the branch condition is true and the branch is taken.

In a running MIPS program, at address 0x8000 0010 there is a `beq` instruction. The binary representation of this instruction is `0x5000 FF00`. The target of this branch is at address:

(a) ( ) `0x8000 FC10`

(b) ( ) `0x8000 FF14`

(c) ( ) `0x7FFF FC10`

(d) ( ) `0x7FFF FF14`

(e) (**X**) `0x7FFF FC14`

d. The windshield in the new WBM35 vehicle has a sensor that detects wether the windshield is fogged. If the visibility is below a set threshold the vehicle computer sets up a timer to wait for the driver to activate the windshield hot air to defog the windshield. If the driver does not activate the hot air and the fog level persists when the timer expires, then the system issues a sound alarm to alert the driver to activate the hot air system. The mechanism used by this system to interact with the driver is called

(a) ( ) polling

(b) ( ) interrupting

(c) (**X**) polling watchdog

(d) ( ) virtual memory

(e) ( ) cached timer

e. When an exception handler starts executing it needs space to store essential information about the program. The exception handler can use registers `$k0` and `$k1` to store such information. If the handler needs more space, it may use:

(a) ( ) caller-saved registers `$t0-$t9`

(b) (**X**) memory allocated in the `.kdata` segment

(c) ( ) memory allocated in the `.data` segment

(d) ( ) space in the program stack

(e) ( ) dynamically allocated memory in the heap

f. The following is **not** an advantage of lazy linkage of library functions

(a) ( ) automatically links the latest version of the library

(b) ( ) only incurs the cost of linking the first time that it invokes a function

(c) ( ) avoids image code bloat

(d) (**X**) can link code that is not relocatable

(e) ( ) only links the code that is actually invoked by each execution of the program

g. *Forwarding* is a technique used in a pipeline to reduce the number of stall cycles that are caused by hazards. Each sequence of instructions shown below cause a hazard for the version of the MIPS pipeline that we studied in class. The pipeline bubble that would be caused by some of these hazards can be avoided through the use of forwarding. Others cannot. Mark the sequences for which the bubble can be avoided through forwarding (**You may need to mark more than one choice in this question**):

(a) ( )
```
lw $t0, 0($t1)
add $t2, $t3, $t0
```

(b) ( )
```
beq $t0, $t1, L1
add $t2, $t3, $t4
```

(c) (X)
```
sub $t0, $t1, $t2
add $t3, $t4, $t0
```

(d) (X)
```
lw $t0, 0($t1)
sll $t5, $t6, $t7
add $t2, $t3, $t0
```

(e) (X)
```
add $t7, $t8, $t9
beqz $t7, L2
```