



### 3. Fill in the blanks

A \_\_\_\_\_ subroutine always has a stack frame associated with it when it is called.

A \_\_\_\_\_ subroutine supports recursion.

A \_\_\_\_\_ subroutine does not (cannot) call other subroutines.

A \_\_\_\_\_ subroutine usually results in a larger code size compared to equivalent code implemented as other types of subroutines.

A \_\_\_\_\_ subroutine shares the same register window set as its caller.

In the SPARC architecture, the \_\_\_\_\_ register points into the current/active stack frame such that \_\_\_\_\_ variables are accessed with a \_\_\_\_\_ offset and \_\_\_\_\_ are accessed with a \_\_\_\_\_ offset.

### 4. Indicate whether the following expressions are

**A.** legal (no compiler error) or

**B.** illegal (compiler error).

int a[10];

int \*iPtr;

a = iPtr; \_\_\_\_\_

iPtr = a; \_\_\_\_\_

\*iPtr = a[3]; \_\_\_\_\_

a[3] = \*iPtr; \_\_\_\_\_

3[a] = 5; \_\_\_\_\_

\*a = iPtr[2]; \_\_\_\_\_

What question would you most like to see on the Midterm?