**REVIEW QUESTIONS**

**1. What is the definition used in this chapter for “simple” subprograms?**

* Subprograms cannot be nested
* All local variables are static

**2. Which of the caller or callee saves execution status information?**

* Either

**3. What must be stored for the linkage to a subprogram?**

* Execution status information

**4. What is the task of a linker?**

* Find files that contain the translated subprograms referenced in that program and load them into memory
* Set the target addresses of all calls to those subprograms in the main program to the entry addresses of those subprograms

**5. What are the two reasons why implementing subprograms with stack-dynamic local variables is more difficult than implementing simple subprograms?**

* Has a more complex activation record. Compiler must generate code to cause implict allocation and deallocation of local variables
* Recursion must be supported ( multiple simulatenous activations of subprograms)

**6. What is the difference between an activation record and an activation**

**record instance?**

* Activation Record = format, layout, non-code part of the subprogram
* Activation Record Instance = collection of data in the form of an activation record. A concrete example of A.R.

**7. Why are the return address, dynamic link, and parameters placed in the**

**bottom of the activation record?**

* These must appear first, since it is a stack

**8. What kind of machines often use registers to pass parameters?**

* RISC machine
* Parameters passed in registers

**9. What are the two steps in locating a nonlocal variable in a static-scoped language with stack-dynamic local variables and nested subprograms?**

* Record instance in the specific stack must be located
* Using the local offset, the record instance can be accessed and inspected

**10. Define static chain, static\_depth, nesting\_depth, and chain\_offset.**

* Static Chain = chain of static links that connect certain activation record instance in the stack
* Static Depth = an integer associated with a static scope that indicates how deeply it is nested in the outermost scope
* Nesting Depth (Chain Offset) = the difference between the static depth of the subprogram containing the reference to X and the static depth of the subprogram containing the declaration for X

**11. What is an EP, and what is its purpose?**

* Points to the base or first address of the A.R. instance of the main program.
* Used to access the parameters and local variables during execution of subprograms

**12. How are references to variables represented in the static-chain method?**

* Represented using **static\_depth**

**13. Name three widely used programming languages that do not allow**

**nested subprograms.**

* C, C++, Java

**14. What are the two potential problems with the static-chain method?**

* Difficult to estimate the costs of nonlocal references which depends on the depth of nesting between the reference and the scope of declaration.
* Code modifications may change nesting depths which changes the timing of some references

**15. Explain the two methods of implementing blocks.**

* Using the static-chain process
* Using a different and somewhat simpler and more efficient way

**16. Describe the deep-access method of implementing dynamic scoping.**

* Deep Access = nonlocal references are found by searching the A.R. instances on the dynamic chain. Length of chain cannot be statically determined every A.R instance must have variable names

**17. Describe the shallow-access method of implementing dynamic scoping.**

* Names and values are stored in a global table
* Space is allocated for every variable name that is in the program.
* When a sub-routine is called, it saves the current value of the variable and replcaes it with the value in its current scope and restores the value of the variable while exiting

**18. What are the two differences between the deep-access method for nonlocal access in dynamic-scoped languages and the static-chain method for static-scoped languages?**

* Scoping: The deep-access method is used in dynamic-scoped languages, where the scope of a variable is determined at runtime. In contrast, the static-chain method is used in static-scoped languages, where the scope of a variable is determined at compile time.
* Access method: The deep-access method uses a linked list of activation records to access nonlocal variables, whereas the static-chain method uses a fixed offset from the base address of the current activation record to access nonlocal variables.

**19. Compare the efficiency of the deep-access method to that of the shallow-access method, in terms of both calls and nonlocal accesses**

* Shallow-access method is more efficient because it involves fewer operations to access variables. It only requires one indirection operation to access a nonlocal variable
* Deep-access method requires multiple indirection operations to traverse the linked list of A.R to find the desired variable