STORAGE ALLOCATION STRATEGIES

STORAGE ALLOCATION STRATEGIES

- The different storage allocation strategies are:
- **Static allocation** - lays out storage for all data objects at compile time
- **Stack allocation** - manages the run-time storage as a stack.
- **Heap allocation** - allocates and deallocates storage as needed at run time from a data area known as heap.

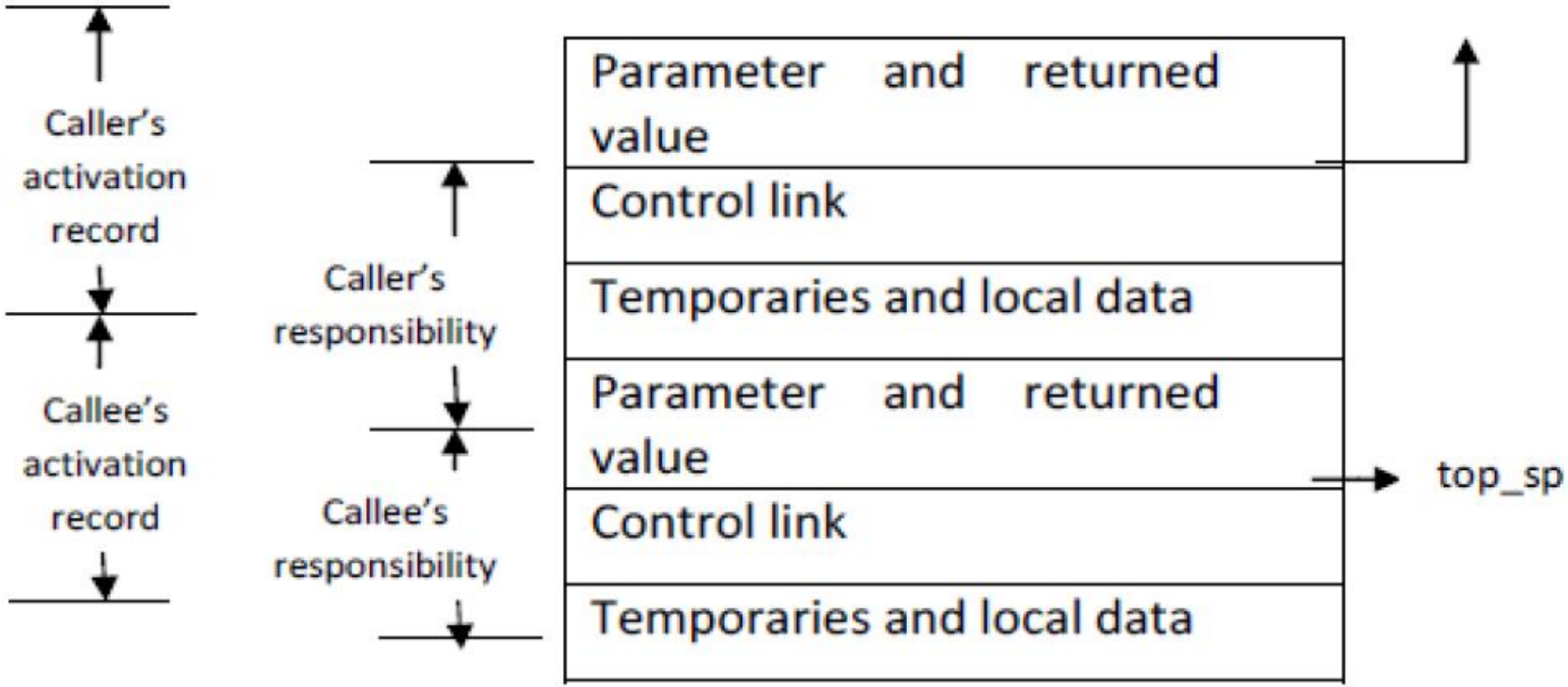
STORAGE ALLOCATION STRATEGIES

- **Static Allocation**
- Here storage allocated at compile time not at run-time or execution time.
- Since the allocation do not change at runtime, every time a procedure activated names bounded to the same storage location allocated at compile time.
- when control returns to a procedure the value of the local are the same as they were when control left the last time.

STORAGE ALLOCATION STRATEGIES

- **Stack Allocation**
- **Here allocation takes place at run time.**
- Each time a procedure called, space for its local variables is pushed onto a stack, and when the procedure terminates, space popped off from the stack.
- Procedures usually implemented as calling sequence & return sequence.
- calling sequence often divided between the calling procedure (caller) and which procedure is called (callee).

STORAGE ALLOCATION STRATEGIES



STORAGE ALLOCATION STRATEGIES

- Value communicated between caller and callee generally placed at the beginning of the caller & callee's activation record, so they as close as possible to the caller's activation record.
- 2. Fixed length items generally placed in the middle. Such items typically include the control link, the access link, and the machine status field.
- 3. Items whose size may not be known early enough placed at the end of the activation record.
- 4. Location of stack top is at the end of fixed length fields in the activation record.

STORAGE ALLOCATION STRATEGIES

- The calling sequence and its division between caller and callee are as follows:

1. The caller evaluates the actual parameters.
2. The caller stores a return address and the old value of top_sp into the callee's activation record.
3. The callee-saves the register values and other status information.
4. The callee initializes its local data and begins execution.

STORAGE ALLOCATION STRATEGIES

- A suitable, corresponding return sequence is:

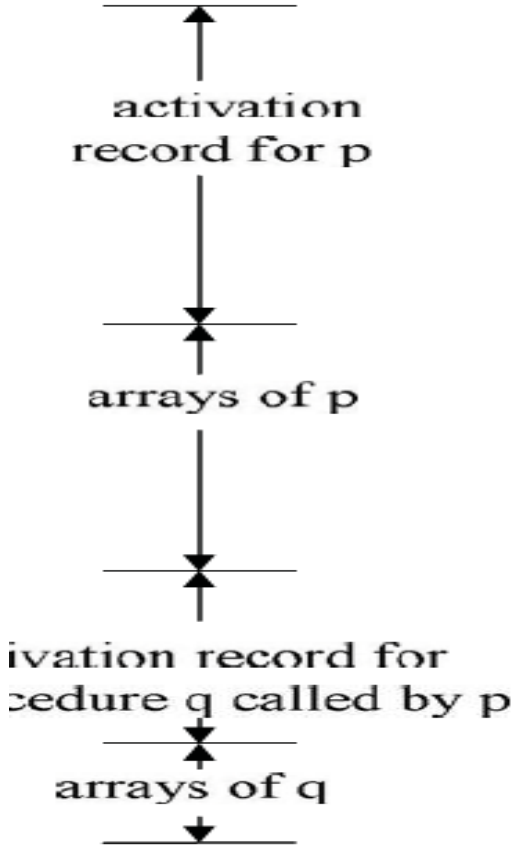
1. The callee places the return value next to the parameters.

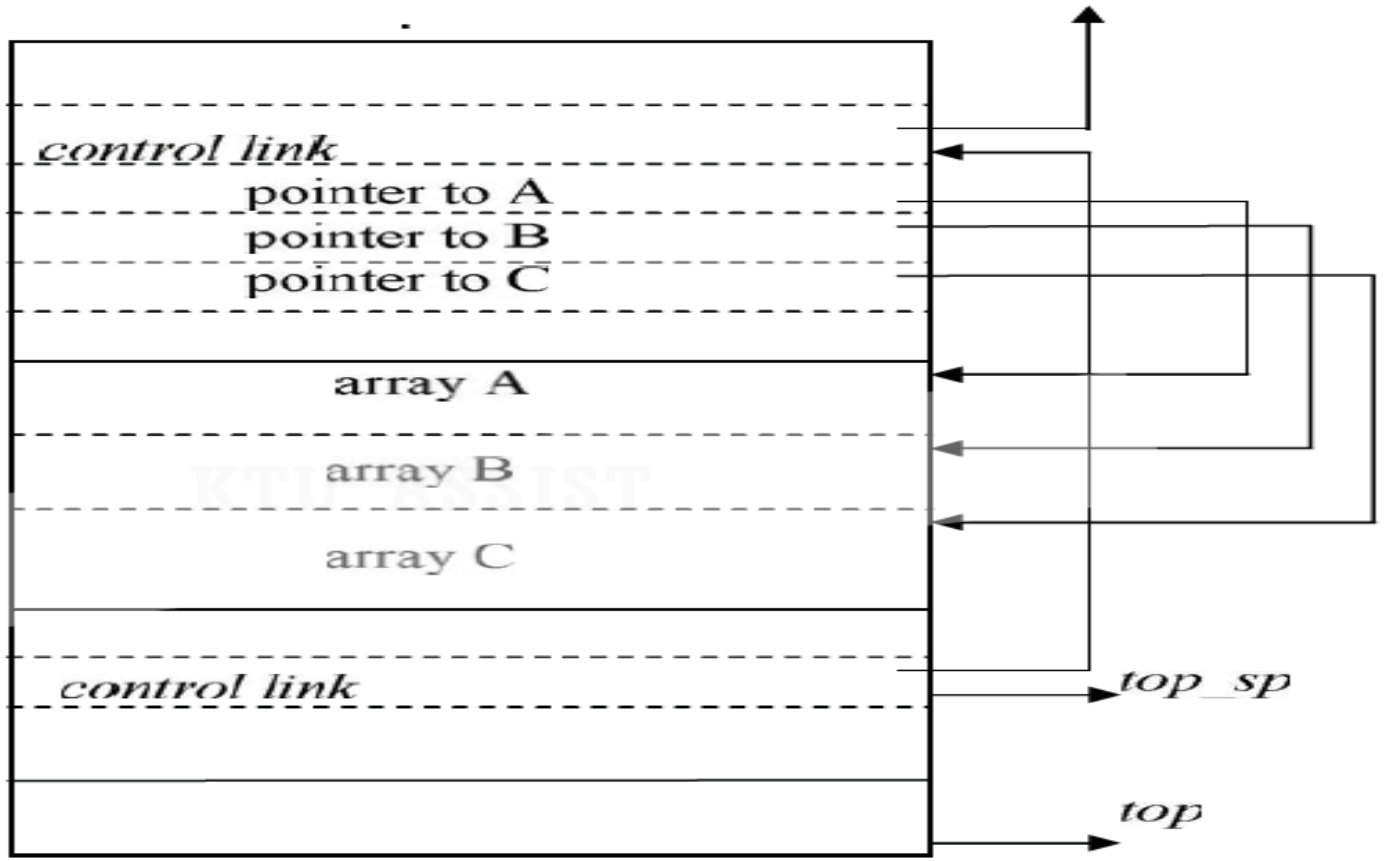
2. Using the information in the machine status field, the callee restores top_sp and other registers, and then branches to the return address that the caller placed in the status field.

3. Although top_sp has been decremented, the caller knows where the return value is, relative to the current value of top_sp; the caller, therefore, may use that value.

STORAGE ALLOCATION STRATEGIES

Variable length data on the stack





STORAGE ALLOCATION STRATEGIES

- objects whose **size cannot be determined** at **compile time** are **allocated** space **in the heap**
- **However**, it is also **possible to allocate objects, arrays, or other structures of unknown size on the stack**.
- **avoid** the **expense of garbage collecting** their space.
- Disadvantage
- Note that the stack can be used only for an object if it is local to a procedure and **becomes inaccessible** when **the procedure returns**.

STORAGE ALLOCATION STRATEGIES

- **Dangling Reference**

- The dangling reference occurs when there is a reference to storage that has been allocated.

Eg: int *k = (int*)malloc(40) //1000 – starting address,used up to 1060

……

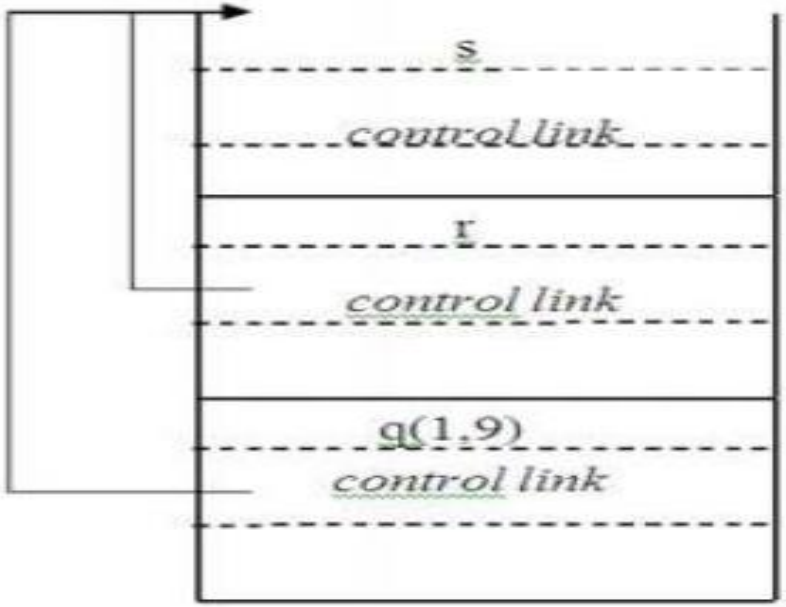
……

free(p); // 1000 to 1060 will be cleared.

*k = 15;

STORAGE ALLOCATION STRATEGIES

- **Heap Allocation**

Position in the activation tree	Activation records in the heap	Remarks
		Retained activation record for r

STORAGE ALLOCATION STRATEGIES

- Stack allocation strategy cannot be used if either of the following is possible :

1. The values of local names must be retained when an activation ends.
2. A called activation outlives the caller

STORAGE ALLOCATION STRATEGIES

- Heap allocation parcels out pieces of contiguous storage, as needed for activation records or other objects.
- Pieces may be deallocated in any order, so over the time the heap will consist of alternate areas that are free and in use.
- The record for an activation of procedure r is retained when the activation ends.
- Therefore, the record for the new activation q(1 , 9) cannot follow that for s physically.
- If the retained activation record for r is deallocated, there will be free space in the heap between the activation records for s and q.