

What is the general non-object-oriented sense of the word "object"?

Anything that might have a name.

What are the available binding times?

- Language design
- Language implementation
- Program-write
- Compile
- Link
- Load
- Run

Name three storage allocation mechanisms.

- Static allocation
- Stack-based allocation
- Heap allocation

What is a static object?

An object given an absolute address that is retained throughout program execution.

Name the two kinds of constants.

How can they be allocated?

- Manifest (AKA compile-time) constants. They can always be allocated statically.
- Elaboration-time. They can refer to run-time values, but cannot be changed once made. When local to recursive subroutines they cannot be statically allocated.

Name two examples of static objects in C

- Global variables in C
- `static` local variables in C

Static allocation of local variables is impossible in languages with ...

Recursion

Good compilers will try to keep arguments, return values, and temporaries where?

In registers.

What is a dynamic link?

A subroutine's reference to the stack frame of the caller.

Each subroutine has its own ...

What does it hold?

... frame (activation record) on the stack, holding arguments, return values, local variables, temporaries, and bookkeeping information.

Who maintains the stack?

The calling sequence of the caller, and the prologue and epilogue of the callee.

What is a calling sequence?

The code executed by the caller immediately before and after a call.

What are the prologue and epilogue?

Code executed at the beginning and at the end of a subroutine.

What is the frame pointer?

What is it useful for?

A register that points to a known location within the frame of the current subroutine.

Code accessing a local variable or argument can do so by predetermined offsets to the frame pointer.

What is displacement addressing?

Hardware support for addressing relative to an offset (like the frame pointer's) inside a normal `load` or `store` instruction.

What is the heap?

It should not be confused with ...

The region of storage in which sub-blocks can be allocated and de-allocated at any time.

Not to be confused with the tree data structure.

What are the major concerns in heap design?

Speed and space. Space can be divided into internal fragmentation and external fragmentation.

What is internal fragmentation?

The assignment of blocks larger than required for a given object. The unused space is lost.

What is external fragmentation?

The result of allocated blocks being scattered throughout the heap. There may be lots of free space, but no one piece of it may be large enough to satisfy a future request.

What is a free list?

A linked list of heap blocks not currently in use.

How does a free list work?

Initially the free list is one block for the whole heap. To satisfy a request some block is given to an object, and some or all of the unused space is returned to the list. When the object is de-allocated neighbor coalescing will be tried.

Name and describe two methods for selecting a heap block for a given object.

- First fit: give object the first block large enough to store it
- Best fit: search list to find the smallest block large enough to store object

Compare the best fit and first fit algorithms.

- Best fit is slower, always searching entire list. It's better at preserving large blocks for large requests but results in many tiny "left-over" blocks.
- Neither is inherently better at reducing external fragmentation. It depends on the distribution of requests.

What is the cost of allocation for memory management algorithms with one free list?

Linear, whether using best of first fit.

How can memory-allocation be reduced to constant time?

How can such techniques be further subdivided?

By maintaining separate free lists ("pools") for blocks of different sizes.

This separation can be done statically or dynamically.

What are two common mechanisms for dynamic pool adjustment?

- Buddy system
- Fibonacci heap

What is the buddy system?

A method of dynamic management of multiple free lists. Standard block sizes are powers of 2. If a block of 2^k is needed but not available, a 2^{k+1} block is split in half, one going to the k th free list. If a split block is de-allocated it coalesces with its buddy, if free.

How does the Fibonacci heap work?

It results in ...

Similarly to the buddy system, but using Fibonacci numbers instead of powers of two for standard block sizes.

... lower internal fragmentation because the Fibonacci sequence grows more slowly than 2^k .

An adversary can always devise a series of memory requests that ...

... cannot be satisfied even though the total space required is less than the size of the heap.

Describe adversarial memory request strategies for static and dynamic allocation schemes.

- Static: exceed the maximum number of requests of a given size that can be satisfied.
- Dynamic: allocate a large number of small objects then de-allocate every other one in order of address, leaving a checkerboard of alternating small free and small allocated blocks.

How can external fragmentation be eliminated?

By "compacting" the heap, which involves moving already-allocated blocks and updating all outstanding references to them.

What are the two main types of manual memory allocation errors?

- Dangling references, i.e. de-allocating too soon. Allows access to memory that has already been de-allocated from the intended object.
- Memory leaks, i.e. de-allocating too late. Not de-allocating an object at the end of its lifetime wastes memory.

What is the scope of a binding?

The textual region in which the binding is active.

What is lexical scoping?

It is also known as ...

A naming system in which it is possible to tell which names refer to which objects at which points in the program based purely on textual rules.

AKA static scoping.

Why is *lexical scope* a better term than *static scope*?

Scope rules based on nesting can be enforced at run-time instead of compile-time, if desired, e.g., Lisp's passing of unevaluated subroutine declaration text.

Define dynamic scoping.

A naming system in which bindings depend on flow of execution at run time.

What is meant by "scope" when no particular binding is in mind?

A program region of maximal size in which no bindings change, or at least none are destroyed.

What is the referencing environment?

The set of active bindings.

What is the closest nested scope rule?

A name introduced in a declaration is known in the scope in which it is declared, and in each internally nested scope unless hidden by another declaration of the same name in one or more nested scopes.

What is a hole in a scope?

A nested scope in which a binding declared in an outer scope cannot be accessed by its normal name because that same name is being used by a different object in the inner-more scope.

Name two methods languages offer for accessing hidden bindings.

- qualifiers
- scope resolution operators

How can subroutines refer to local objects?
To nonlocal ones?

A subroutine can refer to its own objects via displacement addressing on the frame pointer.

It can refer to objects of outermore scopes by traversing the static link chain.

What is a subtlety of static scoping?

Is the scope of a variable the entire block in which it's found, or just the portion after its declaration?

This is a problem when inner scopes hide variables of outer ones.

```
const N = 10
def foo:
    M = N // outer N or semantic error?
    N = 20
```

When is declare-before-use problematic?

How does C/C++ solve the conundrum?

When using recursive types or recursive subroutines.

C/C++ distinguish between declaration and definition, allowing scope to be established by declaration prior to recursive use.

What are forward references?

What is the consequence of allowing them?

Forward references are uses of an object before their declaration. (E.g., Java allows this for class members).

They greatly increase the complexity and memory requirements of a compiler and prevent the compiler from being one-pass.

What are the benefits of information hiding?

- Reduce cognitive load by hiding details.
- Reduce opportunities for name conflicts.
- Safeguard integrity of data abstractions by preventing outside access.
- Compartmentalize run-time errors.

What do the two uses of the C keyword
`static` have in common?

They support a degree of information hiding. One for subroutines and the other for the compilation unit.

What are modules?

Collection of objects encapsulated in such a way that (1) objects inside are visible to one another, but (2) objects on inside are not visible to outside unless explicitly exported, and (3) (in many languages) objects on outside are not visible to objects on inside unless imported.