Loaders

Loaders

- A loader is a system program that performs the loading function.
  - many also support relocation & linking
  - others have a separate linker and loader
- Basic Functions
  - bringing an object program into memory
  - starting its execution

Input

- Object program:
  - contains translated instructions and data from the source program.
  - specifies addresses in memory where these items are to be loaded.

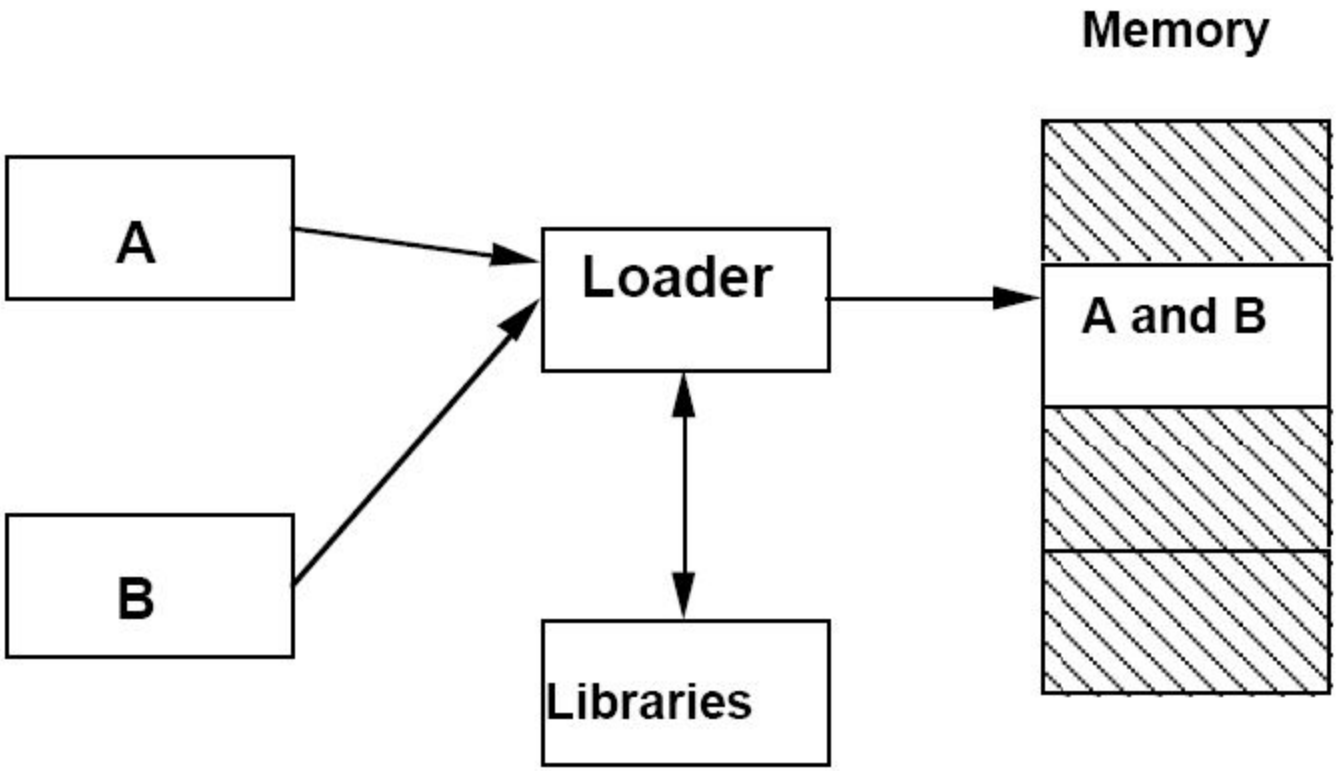
Basic Functions

- ***Allocation***: allocate space in memory for the programs
- ***Linking****:* Resolve symbolic references between object files
  - combines two or more separate object programs
  - supplies the information needed to allow references between them

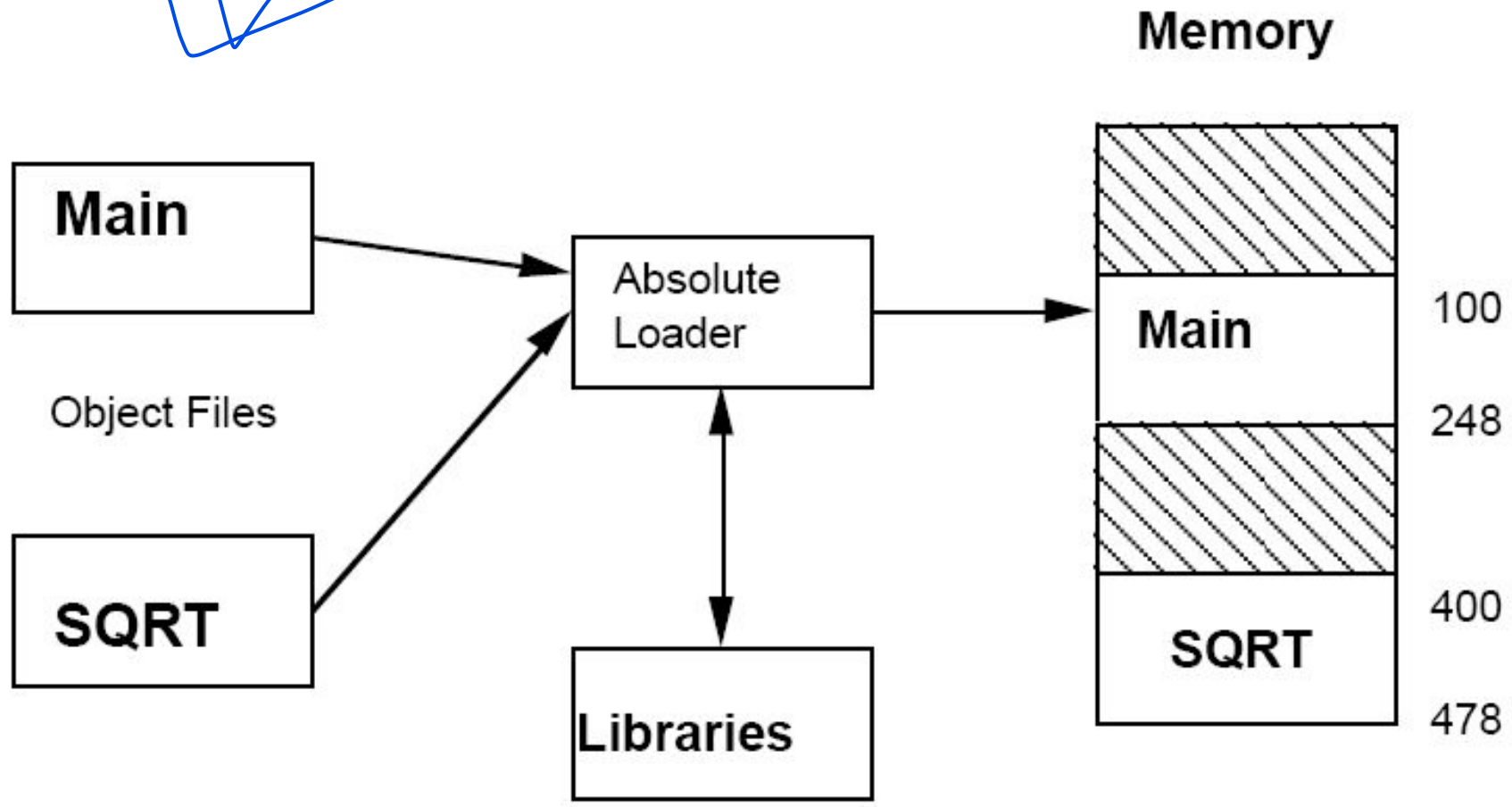
Basic Functions

- ***Relocation:*** Adjust all address dependent locations, such as address constants, to correspond to the allocated space
  - modifies the object program so that it can be loaded at an address different from the location originally specified
- ***Loading:*** Physically place the machine instructions and data into memory

Basic Functions



Absolute Loader



Design of an Absolute Loader

- Its operation is very simple
  - no linking or relocation
- Single pass operation
  - check **H record to verify that correct program** has been presented for loading
  - read each **T record, and move object code into** the indicated address in memory
  - at **E record, jump to the specified address to** begin execution of the loaded program.

Disadvantages

- If changes were made to MAIN that increased its length to more than 300 bytes
  - the end of MAIN (at 100 + 300 = 400) would overlap the start of SQRT (at 400)
  - It would then be necessary to assign SQRT to a new location
    - changing its START and re-assembling it?!
- Furthermore, it would also be necessarily to modify all other subroutines that referred to the address of SQRT.

Disadvantages of Absolute Loaders

- Actual load address must be specified
- The programmer must be careful not to assign two subroutines to the same or overlapping locations
- Difficult to use subroutine libraries (scientific and mathematical) efficiently
  - important to be able to select and load exactly those routines that are needed

Disadvantages of Absolute Loaders

- Allocation - by programmer
- Linking - by programmer
- Relocation - None required-loaded where assembler assigned
- Loading - by loader

Loader Schemes

- Compile and Go
  - The assembler run in one part of memory
  - place the assembled machine instructions and data, as they are assembled, directly into their assigned memory locations
  - When the assembly is completed, the assembler causes a transfer to the starting instruction of the program

Disadvantages

- A portion of memory is wasted because the memory occupied by the assembler is unavailable to the object program.
- It is necessary to re-translate (assemble) the user's program file every time it is run.
- It is very difficult to handle multiple segments, especially if the source programs are in different.

A Simple Bootstrap Loader

- Automatically executed when the computer is first turned on
- Loads the first program to be run: usually the O/S itself begins at address 0 in memory
  - loads the O/S starting at address 80
  - After all code is loaded, bootstrap jumps to address 80.
  - No H or E records, no control information

General Loader Scheme

- Linking
- Relocation
- Loading

Two Pass Direct Linking Loader

- Pass 1
  - **Allocate** and assign each program location in core.
  - Create a symbol table filling in the values of the external symbols.
- Pass 2
  - **Load** the actual program text.
  - Perform the **relocation** modification of any address constants needing to be altered.
  - Resolve external references. (**linking**)

Pass II

- Does actual loading, relocation, and linking

Direct Linking Loader

- The assembler provides
  1. The length of segment
  2. A list of all entries and their relative location within the segment
  3. A list of all external symbols
  4. Information as to where address constants are loaded in the segment and a description of how to revise their values.
  5. The machine code translation of the source program and the relative addresses assigned

Disadvantages of Direct Linking

- It is necessary to allocate, relocate, link, and load all of the subroutines each time in order to execute a program
  - loading process can be extremely time consuming.
- Though smaller than the assembler, the loader absorbs a considerable amount of space
  - Dividing the loading process into two separate programs a binder and a module loader can solve these problems.

Binder

- A binder is a program that performs the same functions as the direct linking loader
  - allocation, relocation, and linking
- Outputs the text in a file rather than memory
  - called a **load module**.
- The module loader merely has to physically load the module into memory.

Binder Classes

- Core image builder:
  - Produces a load module that looks very much like a "snapshot" or "image" of a section of core,
  - Called Core image module.
- Link editor, can keep track of the relocation Information
  - The load module can be further relocated
  - The module loader must perform allocation and relocation as well as loading
  - No linking.

Disadvantage

- If a subroutine is referenced but never executed
  - if the programmer had placed a call statement in the program but was never executed because of a condition that branched around it
  - the loader would still incur the overhead or linking the subroutine.
- All of these schemes require the programmer to explicitly name all procedures that might be called.

Dynamic Loading

- If the total amount of memory required by all subroutines exceeds the amount available
- The module loader loads the only the procedures as they are needed.
  - Allocating an overlay structure
- The Flipper or overlay supervisor is the portion of the loader that actually intercepts the "calls" and loads the necessary procedure.

Dynamic Linking

- The loading and linking of external references are postponed until execution time.
- The loader loads only the main program
- If the main program should
  - execute a branch to an external address,
  - reference an external variable
- The loader is called
  - Only then has the segment containing the external reference loaded.