SYSTEM SOFTWARE

MODULE 3

1. What you mean by relocation of an object program?

   Relocation modifies the object program so that it can be loaded at an address different from the location originally specified.

2. What you mean by linking?

   Linking means combining two or more separate object programs and supplies the information needed to allow references between them.

3. **What you mean by allocation and loading?**

   Allocation and loading means allocating memory location and brings the object program into memory for execution.

4. **What is non relocatable program,relocatable program,self relocating program?**

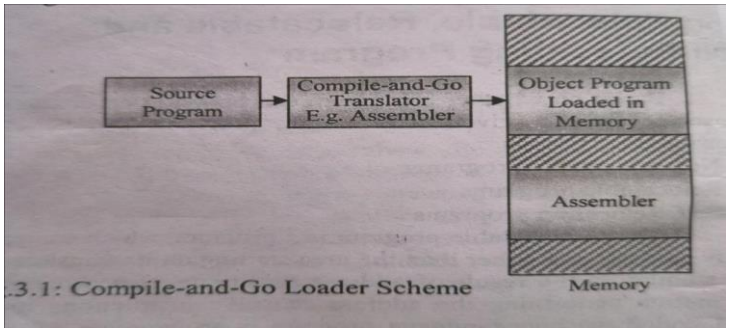
   Non-relocatable programs: A **Non Relocatable program** is one which cannot be made to execute in any area of storage other than the one designated for it at the time of its coding or translation. For example a hand coded machine language program.

   Relocatable program: A program that can be located in different parts of memory at different times. A relocatable program is one that can be processed to relocate it to a desired area of memory. For example, an object module.

   Self-relocating program: A self-relocating program is a program that relocates its own address-dependent instructions and data when run, and is therefore capable of being loaded into memory at any address.

5. **Explain about Compile-and-go loader or Assemble-and-go loader?**

   In this type of loader, the instruction is read line by line, its machine code is obtained and it is directly put in the main memory at some known address. That means the assembler runs in one part of memory and the assembled machine instructions and data is directly put into their assigned memory locations. After completion, the assembly process assigns the starting address of the program to the location counter.



3.1: Compile-and-Go Loader Scheme

Disadvantages

- A portion of memory is wasted because the memory occupied by the assembler or compiler is unavailable to the object program.
- It is necessary to retranslate (assemble) the user's program every time it is run.
- It is very difficult to handle multiple segments, especially if the source programs are in different languages and hence it is difficult to produce modular programs.

Commented [m1]: നോൺ-റിലൊക്കേറ്റബിൾ പ്രോഗ്രാമുകൾ: ഒരു നോൺ റീലോക്കേറ്റബിൾ പ്രോഗ്രാം, അതിന്റെ കോഡിംഗിന്റെയോ വിവർത്തനത്തിന്റെയോ സമയത്ത് അതിനായി നിയുക്തമാക്കിയിട്ടുള്ളതല്ലാതെ മറ്റേതെങ്കിലും സ്റ്റോറേജ് ഏരിയയിൽ എക്സിക്യൂട്ട് ചെയ്യാൻ കഴിയില്ല. ഉദാഹരണത്തിന്, ഒരു ഹാൻഡ് കോഡഡ് മെഷീൻ ലാംഗ്വേജ് പ്രോഗ്രാം.

മാറ്റിസ്ഥാപിക്കാവുന്ന പ്രോഗ്രാം: വ്യത്യസ്ത സമയങ്ങളിൽ മെമ്മറിയുടെ വിവിധ ഭാഗങ്ങളിൽ സ്ഥാപിക്കാൻ കഴിയുന്ന ഒരു പ്രോഗ്രാം. ഒരു റീലൊക്കേറ്റബിൾ പ്രോഗ്രാം എന്നത് ആവശ്യമുള്ള മെമ്മറി ഏരിയയിലേക്ക് മാറ്റാൻ പ്രോസസ്സ് ചെയ്യാവുന്ന ഒന്നാണ്. ഉദാഹരണത്തിന്, ഒരു ഒബ്ജക്റ്റ് മൊഡ്യൂൾ.

സെൽഫ് റീലൊക്കേറ്റിംഗ് പ്രോഗ്രാം: റൺ ചെയ്യുമ്പോൾ സ്വന്തം വിലാസത്തെ ആശ്രയിച്ചുള്ള നിർദ്ദേശങ്ങളും ഡാറ്റയും മാറ്റിസ്ഥാപിക്കുന്ന ഒരു പ്രോഗ്രാമാണ് സെൽഫ് റീലൊക്കേറ്റിംഗ് പ്രോഗ്രാം,

Commented [m2]: ഈ തരത്തിലുള്ള ലോഡറിൽ, നിർദ്ദേശങ്ങൾ വരി വരിയായി വായിക്കുകയും അതിന്റെ മെഷീൻ കോഡ് ലഭിക്കുകയും അത് നേരിട്ട് അറിയപ്പെടുന്ന ഏതെങ്കിലും വിലാസത്തിൽ പ്രധാന മെമ്മറിയിൽ ഇടുകയും ചെയ്യുന്നു. അതായത്, അസംബ്ലർ മെമ്മറിയുടെ ഒരു ഭാഗത്ത് പ്രവർത്തിക്കുന്നു, കൂടാതെ അസംബിൾ ചെയ്ത മെഷീൻ നിർദ്ദേശങ്ങളും ഡാറ്റയും നേരിട്ട് അവരുടെ നിയുക്ത മെമ്മറി ലൊക്കേഷനുകളിൽ ഇടുന്നു. പൂർത്തിയാക്കിയ ശേഷം, അസംബ്ലി പ്രോസസ്സ് പ്രോഗ്രാമിന്റെ ആരംഭ വിലാസം ലൊക്കേഷൻ കൗണ്ടറിലേക്ക് നൽകുന്നു.

Commented [m3]: • അസംബ്ലറോ കംപൈലറോ കൈവശപ്പെടുത്തിയ മെമ്മറി ഒബ്ജക്റ്റ് പ്രോഗ്രാമിന് ലഭ്യമല്ലാത്തതിനാൽ മെമ്മറിയുടെ ഒരു ഭാഗം പാഴാകുന്നു.
• ഓരോ തവണ റൺ ചെയ്യുമ്പോഴും ഉപയോക്താവിന്റെ പ്രോഗ്രാം വീണ്ടും വിവർത്തനം ചെയ്യേണ്ടത് (അസംബ്ലിംഗ്) ആവശ്യമാണ്.
• ഒന്നിലധികം സെഗ്മെന്റുകൾ കൈകാര്യം ചെയ്യുന്നത് വളരെ ബുദ്ധിമുട്ടാണ്, പ്രത്യേകിച്ചും സോഴ്സ് പ്രോഗ്രാമുകൾ വ്യത്യസ്ത ഭാഷകളിലാണെങ്കിൽ മോഡുലാർ പ്രോഗ്രാമുകൾ നിർമ്മിക്കുന്നത് ബുദ്ധിമുട്ടാണ്.

6. Explain about general loader schemes

General Loader Scheme

- The loader accepts the assembled machine instruction, data and other information present in object format and places machine instruction and data in core in an executable computer form.
- The loader is assumed to be smaller so that more memory is available to user.
- Reassembly is no longer necessary to run the program at later stage.

GENERAL LOADER SCHEME

- In "Compile-and-Go" the outputting instruction and data are assembled. In which assembler is placed in main memory that results in wastage of memory.
- To overcome that we requires the addition of the new program of the system, a loader.
- Generally the size of loader is less than that of assembler.

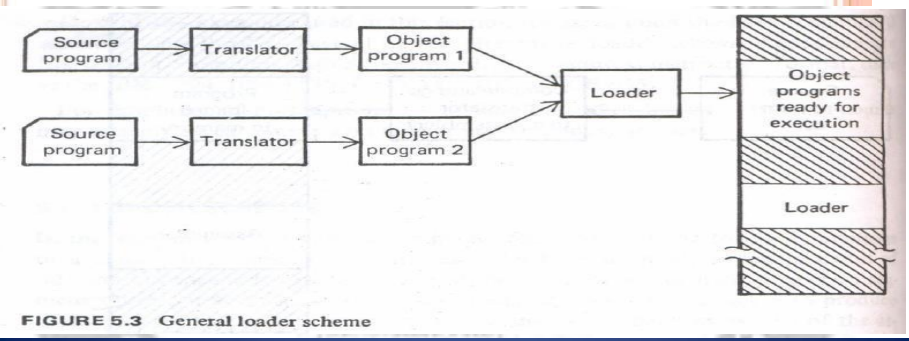


FIGURE 5.3 General loader scheme

GENERAL LOADER SCHEME

- Uses an object deck as intermediate data.
- Here the loader accepts the assembled machine instructions and data in the object format and places it in the memory in an executable form.

ADVANTAGES :

- Smaller than assembler.
- No reassembly is needed.
- Possible to write subroutines in different languages.

7. Explain about Absolute loader or Bootstrap loader?

The absolute loader is a kind of loader in which relocated object files are created, loader accepts these files and places them at a specified location in the memory. This type of loader is called absolute loader because no relocating information is needed; rather it is obtained from the programmer or assembler. The starting address of every module is known to the programmer, this corresponding starting address is stored in the object file then the task of loader becomes very simple that is to simply place the executable form of the machine instructions at the locations mentioned in the object file.

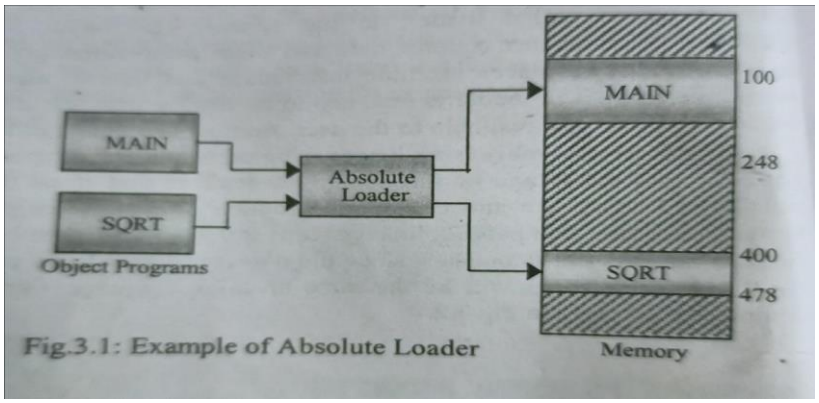


Fig.3.1: Example of Absolute Loader

Advantages:

- It is simple to implement.
- This scheme allows multiple programs or the source programs written in different languages. If there are multiple programs written in different languages then the respective language assembler will convert it to the language and common object file can be prepared.
- The task of loader becomes simpler as it simply obeys the instruction regarding where to place the object code to the main memory.
- The process of execution is efficient.

Disadvantages:

- In this scheme, it's the programmer's duty to adjust all the inter-segment addresses and manually do the linking activity. For that, it is necessary for a programmer to know the memory management.
- If at all any modification is done to some segment the starting address of immediate next segments may get changed the programmer has to take care of this issue and he/she needs to update the corresponding starting address on any modification in the source.

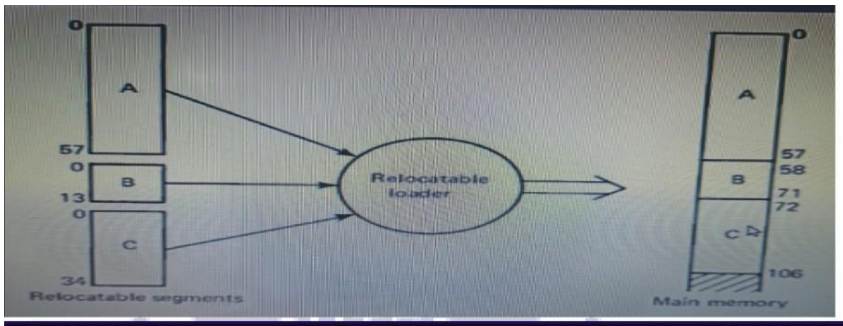
8. Explain about relocating loader or relative loader?

A *relocating loader* is capable of loading a program to begin anywhere in memory. The execution of the object program is done using any part of the available and sufficient memory. The object program is loaded into memory wherever there is room for it.

Commented [m4]: സമ്പൂർണ്ണ ലോഡർ എന്നത് മാറ്റിസ്ഥാപിച്ച ഒബ്ജക്റ്റ് ഫയലുകൾ സൃഷ്ടിക്കുന്ന ഒരു തരം ലോഡറാണ്, ലോഡർ ഈ ഫയലുകൾ സ്വീകരിക്കുകയും മെമ്മറിയിൽ ഒരു നിർദ്ദിഷ്ട സ്ഥലത്ത് സ്ഥാപിക്കുകയും ചെയ്യുന്നു. ഈ തരത്തിലുള്ള ലോഡറിനെ കേവല ലോഡർ എന്ന് വിളിക്കുന്നു, കാരണം സ്ഥലം മാറ്റുന്ന വിവരങ്ങളൊന്നും ആവശ്യമില്ല; മറിച്ച് പ്രോഗ്രാമറിൽ നിന്നോ അസംബ്ലറിൽ നിന്നോ ആണ് ലഭിക്കുന്നത്. ഓരോ മൊഡ്യൂളിന്റെയും ആരംഭ വിലാസം പ്രോഗ്രാമർക്ക് അറിയാം, ഈ അനുബന്ധ ആരംഭ വിലാസം ഒബ്ജക്റ്റ് ഫയലിൽ സംഭരിച്ചിരിക്കുന്നു, തുടർന്ന് ലോഡറിന്റെ ചുമതല വളരെ ലളിതമാണ്, അതായത് ഒബ്ജക്റ്റ് ഫയലിൽ സൂചിപ്പിച്ചിരിക്കുന്ന സ്ഥലങ്ങളിൽ മെഷീൻ നിർദ്ദേശങ്ങളുടെ എക്സിക്യൂട്ടബിൾ ഫോം സ്ഥാപിക്കുക. .

Commented [m5]: • ഇത് നടപ്പിലാക്കാൻ ലളിതമാണ്.
• ഈ സ്കീം ഒന്നിലധികം പ്രോഗ്രാമുകൾ അല്ലെങ്കിൽ വിവിധ ഭാഷകളിൽ എഴുതിയ സോഴ്സ് പ്രോഗ്രാമുകൾ അനുവദിക്കുന്നു. വിവിധ ഭാഷകളിൽ ഒന്നിലധികം പ്രോഗ്രാമുകൾ എഴുതിയിട്ടുണ്ടെങ്കിൽ, അതത് ഭാഷാ അസംബ്ലർ അത് ഭാഷയിലേക്ക് പരിവർത്തനം ചെയ്യുകയും പൊതുവായ ഒബ്ജക്റ്റ് ഫയൽ തയ്യാറാക്കുകയും ചെയ്യും.
• ഒബ്ജക്റ്റ് കോഡ് മെയിൻ മെമ്മറിയിലേക്ക് എവിടെ സ്ഥാപിക്കണം എന്നതിനെക്കുറിച്ചുള്ള നിർദ്ദേശങ്ങൾ ലളിതമായി അനുസരിക്കുന്നതിനാൽ ലോഡറിന്റെ ചുമതല ലളിതമാകുന്നു.
• നിർവ്വഹണ പ്രക്രിയ കാര്യക്ഷമമാണ്.
ദോഷങ്ങൾ:
• ഈ സ്കീമിൽ, എല്ലാ ഇന്റർ-സെഗ്മെന്റ് വിലാസങ്ങളും ക്രമീകരിക്കുകയും ലിങ്കിംഗ് പ്രവർത്തനം നേരിട്ട് നടത്തുകയും ചെയ്യേണ്ടത് പ്രോഗ്രാമറുടെ കടമയാണ്. അതിനായി ഒരു പ്രോഗ്രാമർക്ക് മെമ്മറി മാനേജ്മെന്റ് അറിയേണ്ടത് അത്യാവശ്യമാണ്.
• ഏതെങ്കിലും സെഗ്മെന്റിൽ എന്തെങ്കിലും മാറ്റം വരുത്തിയാൽ, അടുത്ത സെഗ്മെന്റുകളുടെ ആരംഭ വിലാസം മാറിയേക്കാം, പ്രോഗ്രാമർ ഈ പ്രശ്നം ...

Commented [m6]: മെമ്മറിയിൽ എവിടെയും ആരംഭിക്കുന്നതിന് ഒരു പ്രോഗ്രാം ലോഡ് ചെയ്യാൻ ഒരു റീലോക്കേറ്റിംഗ് ലോഡറിന് കഴിയും. ലഭ്യമായതും മതിയായതുമായ മെമ്മറിയുടെ ഏതെങ്കിലും ഭാഗം ഉപയോഗിച്ചാണ് ഒബ്ജക്റ്റ് പ്രോഗ്രാമിന്റെ എക്സിക്യൂഷൻ ചെയ്യുന്നത്. ഒബ്ജക്റ്റ് പ്രോഗ്രാം മെമ്മറിയിലേക്ക് ലോഡുചെയ്യുന്നു, അതിന് ഇടമുള്ളിടത്തെല്ലാം.



- To avoid possible reassembling of all subroutines when a single smlor ml we is cla n g ecl a eel to perf orm th e ta sks of allocation and linking for the programmer the relocating loaders is introd .
- The exec w ti or of t be oDje ct p rogra is clane psi ng any part of theavailable suf cient memory.
- The object program is loaded into memory wherever there is room for it

RELOCATING LOADER

- **ADVANTAGES:**
- The relocating loader takes of the program modification dynamically (i.e change n program s ze)
- Using the transfer vectors,the required sub-routines can be brought into the main memory rather than keeping all routines in the memory all the time .

DISADVANTAGES:

- Transfer vector nkage is useful only for subroutines but not for loading or storing external data.
- If Transfer vector itself increases, the size of object program in memory increases, as Transfer Vector has to be there in memory all the time

9. Explain about direct linking loader?

Direct Linking Loaders

- A Direct linking loader is a general relocating loader and is the most popular loading scheme presently used.
- This scheme has an advantage that it allows the programmer to use multiple procedure and multiple data segments.
- In addition, the programmer is free to reference data or instructions that are contained in other segments.
- The direct linking loaders provide flexible intersegment referencing and accessing ability.

translation of modules. In direct-linking loading scheme, the assembler (translator) must give the loader the following information with each procedure or data segment.

- The length of the segment
- A list of all symbols in the segment that may be referenced by other segments and their relative location within the segment
- A list of all symbols not defined in the segment but referenced in the segment.
- Information as to where address constants are located in the segment and a description of how to revise their values
- 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.

10. Explain about binders and module loaders?

One disadvantage of direct linking loader is that it is necessary to allocate, relocate, link and load all of the modules each time in order to execute a program. So the loading process can be extremely time consuming. And the loader program may be smaller than a translator; it does absorb a considerable amount of memory space. These problems can solve by dividing the loading process into 2 separate programs: **a binder and a module loader.**

Binder is a program that performs the function as direct linking loader in binding together.

It outputs the text as a **file**, rather than placing the relocated and linked text directly into memory. The output files are in format ready to be loaded and are called **a load module. The module loader** loads the module into memory. The binder performs the function of the allocation, relocation and linking. The ***module loader*** performs the function of loading.

11. Explain about the two major classes of binders?

There are 2 major classes of binders

Core image builder: A specific memory allocation of the program is performed at a time that the subroutines are bound together. It is called a core image module and the corresponding binder is called a core image builder

Advantages: Simple to implement and fast to execute.

Disadvantages: Difficult to allocate and load the program.

Linkage editors: The linkage editor can keep track of relocation information so that the resulting load module can be further relocated and the module loader must performs additional allocation and relocation as well as loading but it does not worry about the problem of linking.

Advantages: More flexible allocation and loading scheme **Disadvantages**:

Implementation is so complex.

12. Explain about overlays?

In a general computing sense, overlaying means "the process of transferring a block of program code or other data into main memory, replacing what is already stored".

Overlaying is a programming method that allows programs to be larger than the computer's main memory.

A program containing overlays is called an **overlay-structured program.**

An overlay-structured program consists of:-

A permanently resident portion, called **root** and A
set of **overlays**.

Execution of an overlay-structered program proceedes as follows:-

To start with, the root is loaded in the memory and given control for the purpose of execution.

Other overlays are loaded as and when necessary.

Commented [m7]: ഡയറക്ട് ലിങ്കിംഗ് ലോഡറിന്റെ ഒരു പോരായ്മ, ഒരു പ്രോഗ്രാം എക്സിക്യൂട്ട് ചെയ്യുന്നതിനായി ഓരോ തവണയും എല്ലാ മൊഡ്യൂളുകളും അനുവദിക്കുകയും മാറ്റി സ്ഥാപിക്കുകയും ലിങ്ക് ചെയ്യുകയും ലോഡ് ചെയ്യുകയും ചെയ്യേണ്ടത് ആവശ്യമാണ്. അതിനാൽ ലോഡിംഗ് പ്രക്രിയ വളരെ സമയമെടുക്കും. കൂടാതെ ലോഡർ പ്രോഗ്രാം ഒരു വിവർത്തകനേക്കാൾ ചെറുതായിരിക്കാം; ഇത് ഗണ്യമായ അളവിൽ മെമ്മറി സ്പേസ് ആഗിരണം ചെയ്യുന്നു. ലോഡിംഗ് പ്രക്രിയയെ 2 വ്യത്യസ്ത പ്രോഗ്രാമുകളായി വിഭജിച്ച് ഈ പ്രശ്നങ്ങൾ പരിഹരിക്കാൻ കഴിയും: ഒരു ബൈൻഡറും ഒരു മൊഡ്യൂൾ ലോഡറും.

Commented [m8]: റീലൊക്കേറ്റ് ചെയ്തതും ലിങ്ക് ചെയ്തതുമായ ടെക്സ്റ്റ് നേരിട്ട് മെമ്മറിയിലേക്ക് സ്ഥാപിക്കുന്നതിനുപകരം ഇത് ടെക്സ്റ്റിനെ ഒരു ഫയലായി ഔട്ട്പുട്ട് ചെയ്യുന്നു. ഔട്ട്പുട്ട് ഫയലുകൾ ലോഡുചെയ്യാൻ തയ്യാറായ ഫോർമാറ്റിലാണ്, അവയെ ലോഡ് മൊഡ്യൂൾ എന്ന് വിളിക്കുന്നു. മൊഡ്യൂൾ ലോഡർ മൊഡ്യൂളിനെ മെമ്മറിയിലേക്ക് ലോഡ് ചെയ്യുന്നു. അലോക്കേഷൻ, റീലോക്കേഷൻ, ലിങ്കിംഗ് എന്നിവയുടെ പ്രവർത്തനം ബൈൻഡർ നിർവഹിക്കുന്നു. മൊഡ്യൂൾ ലോഡർ ലോഡിംഗിന്റെ പ്രവർത്തനം നിർവ്വഹിക്കുന്നു.

The loading of an overlay overwrites a previously loaded overlay with the same load origin.

This reduces the memory requirement of a program.

13. Explain about dynamic loading using overlays?

We have assumed that all of the modules needed are loaded into the memory at the same time. If the total amount of memory required by all of these modules exceeds the amount of available memory, the program cannot execute. This problem can be avoided by dynamic loading based on overlays. Usually the modules of a program are needed at different times. By explicitly recognizing which modules call other modules it is possible to produce an overlay structure that identifies manually exclusive modules.

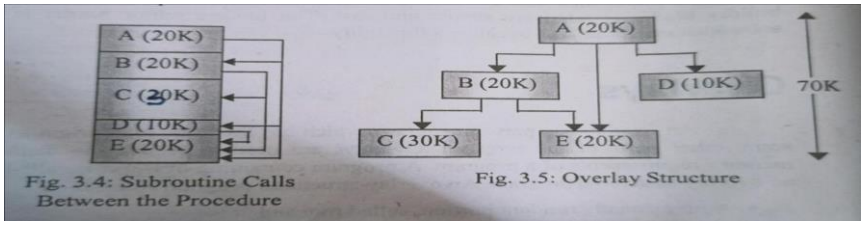


Fig. 3.4: Subroutine Calls Between the Procedure

Fig. 3.5: Overlay Structure

14. Explain about dynamic binding or linking?

A major disadvantage of all the previous loading scheme is that if a module is referenced but never executed, the loader will still incur the overhead of linking the module. In dynamic linking scheme, linking and loading of external references are postponed until execution time. To start with, the loader loads only the main module. If the main module references an external reference, the loader is called. Only then is the segment containing the external reference loaded.

An advantage of this scheme is that no overhead is incurred unless the module to be called is actually used.

The major disadvantage of this scheme is the considerable overhead and complexity incurred, due to the postponement of most of the binding process until execution time.

15. Explain about design of absolute loader?

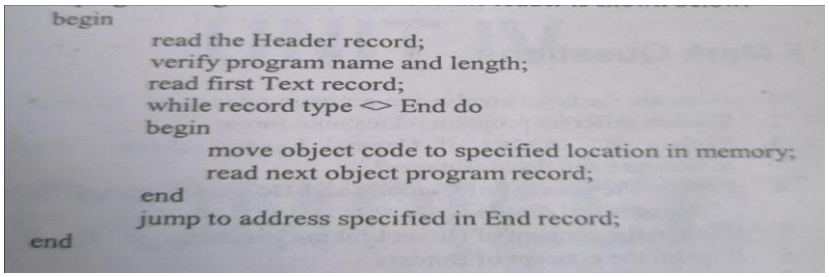
Absolute loaders are the loaders that do not perform any relocation or linking. Loader read the object program and moves the text in the object program into the absolute locations specified by the translator. All functions are accomplished in a single pass.

Object program contain 3 types of records: Header, Text, End

Header:- Program name, starting address and length.

Text:- translated instructions and data of the program and the address where these are to be loaded.

End:- indicates end of the program and specifies the address in the program where execution is to begin.



```
begin
        read the Header record;
        verify program name and length;
        read first Text record;
        while record type <> End do
        begin
                move object code to specified location in memory;
                read next object program record;
        end
        jump to address specified in End record;
end
```