Assignment No. 1

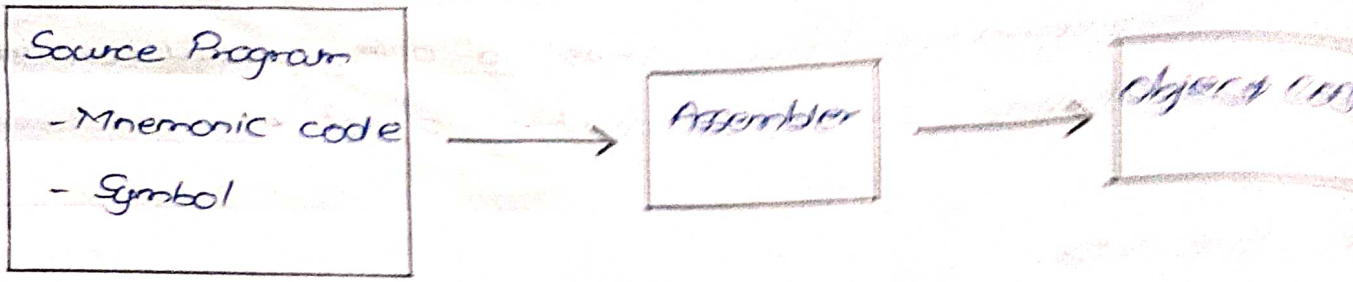
* Title:
  - Design suitable data structures and implement pass-I assembler for pseudo machine in Java using object oriented feature.
  - Implementation should consist of a few instruction from each category and few assembler directives.

* Objectives:
  - To understand data structure of pass-I assembler.
  - To understand pass-I assembler concept.
  - To understand Advanced Assembler Directives.

* Problem Statement:
  - Design suitable data structures and implement pass-I of a two-pass assembler for pseudo-m/c in Java using object oriented feature.

* Outcomes:
  - Students will be able to
    1) Implement pass-I assembler.
    2) Implement Symbol table, Literal table and pool tabl
    3) Understand concept of Advanced Assembler Directive.

* Software / Hardware Requirements:
  - latest JDK, eclipse.
  - M/c Lenovo Think Center M700 Ci3, 6100, 6th gen, H81, 4GB RAM, 500GB HDD

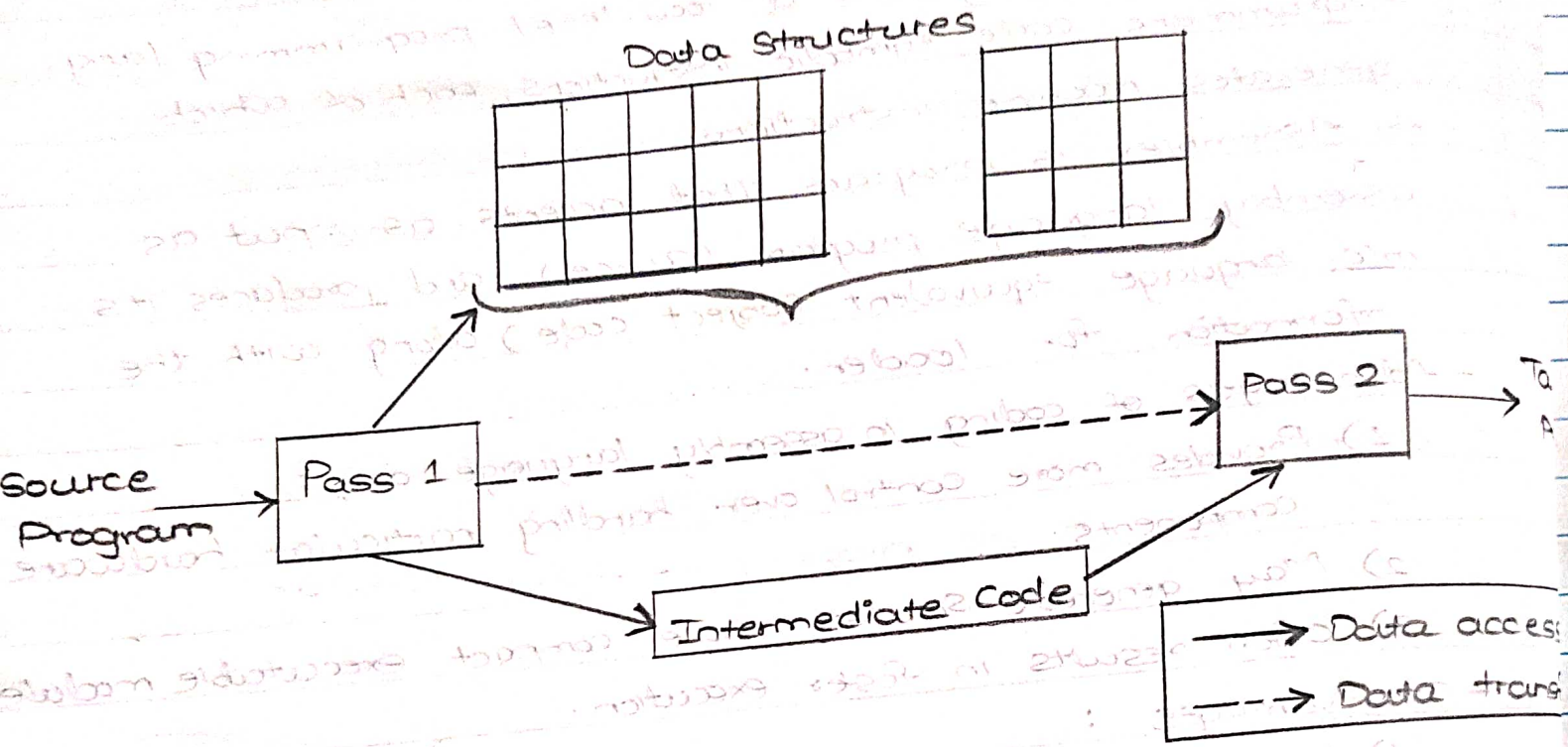


* fig: Executable Program Generation from an assembly source code *

* Theory:

- There are 2 main classes of programming languages: High-level (Eg C, pascal) and Low level.
- Assembly Language is a low-level programming lang.
- Programmers code symbolic instructions, each of which generates machine instructions.
- An Assembler is program that accepts as input as assembly language program (source) and produces its m/c language equivalent (object code) along with the information for loader.
- Advantages of coding in assembly language are:
  1) Provides more control over handling particular hardware components.
  2) May generate smaller, more compact executable modules.
  3) Often results in faster execution.
- Disadvantages:
  1) Not Portable.
  2) More Complex.
  3) Requires understanding of hardware details

- Pass - I Assembler:
- An Assembler does the following:
  1) Generate machine Instruction:
     - Evaluate mnemonics to produce their m/c code.
     - Evaluate symbols, literals, addresses to produce their equivalent m/c. addresses.
     - Convert data constants into their m/c representation.
  2) Process pseudo operations.



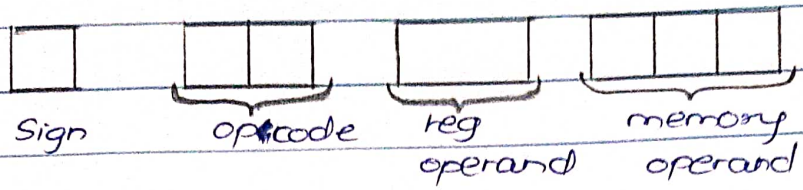
* fig: A simple two pass assembler *

- Pass- 2 Assembler :
  - A two-pass assembler performs two sequential scans over the source code :
    Pass 1 : Symbols and Literals are defined
    Pass 2 : Object program is generated.
- Parsing : Moving in program lines to pull out op-codes and operands.

- Data structures :
  1) Location Counter (LC):
     - Points to the next location where code will be placed.
  2) Symbol table :
     - Contains labels and their values.
  3) String storage Buffer (SSB) :
     - Contains ASCII characters for the strings.
  4) Forward Reference table (FRT):
     - Contains pointer to the String in SSB and offset where its value will be inserted in object code.
  5) Opcode transition table:
     - Contains symbolic instructions, their lengths and their opcode

- Elements of Assembly Language :
  1) Mnemonics Operation Codes.
  2) Symbolic Operands.
  3) Data declarations.
- Statement format :
  [Label] <opcode> <operand spec> [, operand spec> ..]
where,
  [..] indicates that enclosed specification is optional.

- Instruction format :



- Sign is not part of Instruction.

- Assembly Language Statements :
  - Three Kinds of Statements :
    1) Imperative Statements.
    2) Declaration Statements.
    3) Assembler Statements.

- Pass Structure of Assemblers :
  - Two types
    1) Single Pass Assembler.
    2) Two pass Assembler.

* Conclusion :
- Thus, I have implemented pass-I assembler with symbol table, literal table and pool table.

Ashwini
(39)

* Title :
- Implement pass-II of two pass assembler for pseudo-machine in Java using object oriented features.
- The output of assignment-1 should be input for this assignment.

* Objectives :
- To understand datastructures of pass 1 and pass 2 assembler.
- To understand pass 1 and pass 2 assembler output.
- To understand Advanced Assembler Directives.

* Problem Statement :
- To Implement pass-II of two pass assembler for pseudo-machine in Java using object oriented features.
- The output of assignment-1 should be input for this assignment

* Outcomes :
- Students will be able to
  1) Implement pass-2 assembler.
  2) Implement machine-code from intermediate code.
  3) Understood concept pass-2 assembler.

* S/w / H/w requirements :
- Latest JDK, eclipse.
- M/c Lenovo think centre M700 CPi3, 6100, 6th Gen, H81, 4GB RAM, 500 GB HDD.

* Theory :
- There are two main classes of programming languages
- ~~There~~ 1) High Level (E.g. C, Pascal)
  2) Low Level (E.g. Assembly Language)

Teacher's Sign.: _____

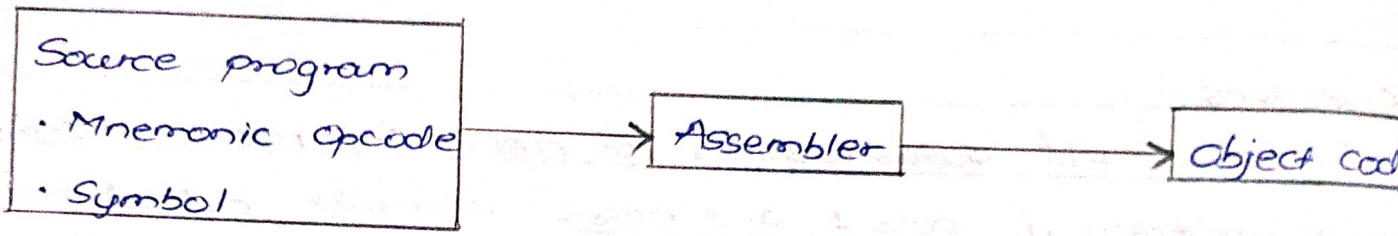
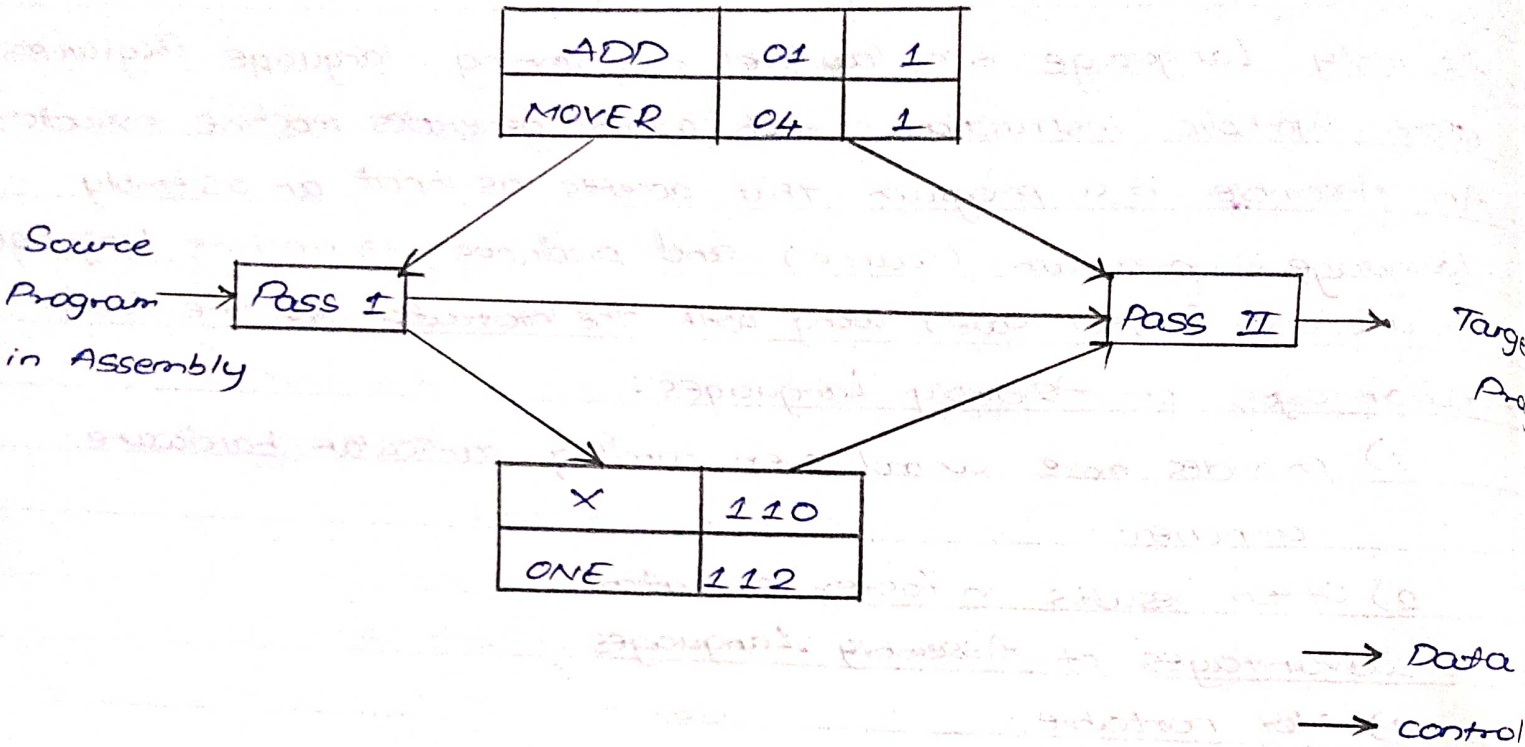


fig: Executable program generation from an assembly source code.

- Assembly Language is a low level programming language. Programers code symbolic instructions, each which generates machine instructions
- An Assembler is a program that accepts as input an assembly language program (source) and produces its machine language equivalent (object code) along with the information for the loader.
- Advantages of Assembly Languages:
  1) Provides more control over handling particular hardware component.
  2) Often results in faster execution.
- Disadvantages of Assembly Languages:
  1) Not portable.
  2) More Complex.
  3) Requires understanding of Hardware details

- Pass-I : (Analysis of Source Program)
  1) Separate the symbol, mnemonics opcode and operand fields.
  2) Build the symbol table.
  3) Perform LC processing.
  4) Construct intermediate representation.

- Pass-II :
  - Processes the intermediate representation (IR) to synthesize the target program.



* fig: Data Structures for assembler *

Page No.
Date

* Conclusion :

- Thus, I have implemented Pass-2 assembler by taking input as output of assignment - I

Page No. A3
Date

* Title :
- Design suitable data structure and implement pass-I of a two-pass macro-processor using OOP features in Java.

* Objectives :
- To understand Data structure of pass-1 macroprocessor
- To understand pass-1 macroprocessor concept.

* Problem Statement :
- Design suitable data structure and implement pass-I of two-pass macro-processor using OOP features in Java.

* Outcomes :
- Students will be able to
  - Implement pass-1 macroprocessor
  - Concept of Pass-1 macroprocessor

* Software / Hardware Requirement :
- JDK / Eclipse.
- M/C Lenovo Think Center M700, Ci3, 6200, 6th gen, H81, 4GB RAM, 500GB HDD.

Theory :
- Macroprocesor :
  - Macro pre-processor takes a source program containing macro definitions and macro calls and translates into an assembly language program. without any macro definition or calls.
  - This program is now handed overed to target program.

Teacher's Sign.: _____

- Macro :
  - Macro allows a sequence of source language code to be defined once and then referred to by name each time it is refered.
  - Each time this name occurs in a program, the sequence of code is substituted at that point.
  - Macro has following parts :
    1) Name of macro.
    2) Parameters in macro.
    3) Macro definition.

- Defining a macro :

macro
  macro name
    macro body

  End of macro definition

- for example

```
MACRO  mymacro
  ADD  AREG, X
  ADD  BREG, X
  MEND.
```

- 'MACRO' pseudo-op is the first line of definition and identifies following lines as macro instruction name.
- Following name line is sequence of instruction being abbreviated the instructions comprising 'MACRO' instruction.
- The definition is terminated by 'MEND' pseudo-op

Teacher's Sign.: _____

- How to call a macro ...?
  - Macro is called by writing a name with actual parameters
  - Syntax
    <macro-name> [<list of parameters>]
  - Example:
    - for above definitions of macro
      1) mymacro
      2) addmacro ×

- Macro Expansion
  - Each call to macro is replaced by its body.
  - During replacement, actual parameters is used in place of formal parameters.
  - During macro expansion, each statement forming the body of macro is picked up one by one sequentially.
  - Formal parameters are replaced by actual parameter value.

- Macro with keyword parameter
  - These are the methods to call macro with formal parameters
  - There are 2 types
    1) Positional parameters:
       - Initiated with '&'
    2) Keyword parameters:
       - Initiated with '&' but has some default value

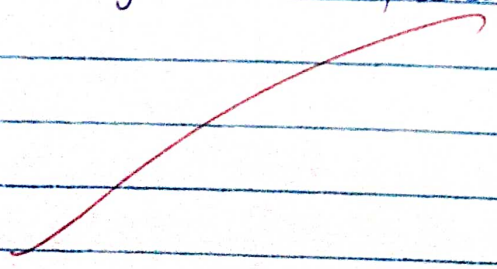
Teacher's Sign.: _____

- Nested macro call
  - Nested macro calls are just like function calls in our normal calls.
  - Only the transfer of control from one macro to another is done.
  - for example:

```
MACRO  innermacro
       ADD AREG, X
MEND
MACRO  outermacro
       innermacro
       ADD AREG, Y
MEND

outermacro
```

- Data structures of two-pass macros
  1) Macro Name Table Pointer (MNTP)
  2) Macro Definition Table Pointer (MDTP)
  3) Macro Name Table (MNT)
  4) Macro Definition Table (MDT)
  5) Argument List Array (ALA)
     - Index given to parameters



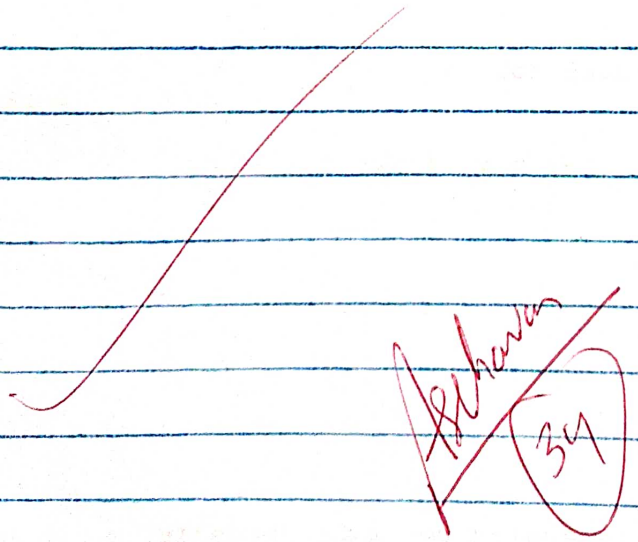
Page No.
Date

* Algorithm:

- Scan all MACROS definitions one by one sequentially
  1) Enter its name in Macro Name Table (MNT)
  2) Store entire macro definition in Macro Definition Table (MDT).
  3) Add the information in MNT indicates where definition of macro can be found in MDT.
  4) Prepare argument list ~~of~~ array (ALA)

* Conclusion:

- Hence, we have implemented pass-I macroprocessor by producing MNT and MDT.



* Title :
- Implement pass - II of two-pass macroprocessor

* Objectives :
- To implement pass-II of two-pass macroprocessor
- To understand concept of pass-II macroprocessor

* Problem Statement :
- Write a java program for pass-II of two-pass macro-processor
- Output of assignment-3 should be input for this.

* Outcomes :
- Students will be able to
- understand concept of pass-II of macroprocessor
- Implement pass-II macroprocessor.

* S/W or H/W requirements :
- JDK / Eclipse.
- M/C Lenovo Think Center 6th gen, 4GB RAM, 500 GB HDD

* Theory :
- Advanced macro facilities.
1) AIF :
- Use of AIF instruction to branch according to the results of a condition test.
- Provides loop control for conditional assembly processing
- Also let you check for errors conditions

2) AGO :

- An AGO instruction branches unconditionally.
- You can thus alter the sequence in which you assembler languages statement processed.
- This provides you with final exits from conditional assembly loops.

3) Sequence Symbols :

- You can use Sequence Symbol in the name field of a statements to branch to that statements during conditionally assembly processing, thus altering sequence in which the assembler processes your conditional assembly and macro instruction.
- It consists of a period (.) followed by alphabetic characters, followed by 0 to 61 alphanumeric characters

4) Expansion Time variables

- Data structures of pass-II macroprocessor

1) Input Source program for pass-II. It is produced by pass-I
2) Macro Definition Table (MDT) produced by pass-I
3) Macro Name Table (MNT) produced by pass-I
4) Macro Name Table Pointer (MNTP) produced by pass-I.
5) Argument List Array (ALA)
6) Source program with macro-calls expanded. This gives output of pass-II
7) Macro Definition Table pointer (MDTP).

Teachers Sign: _____

Page No.
Date

* Algorithm:

- Take input from pass - I
- Examine all statements in assembly source program to detect macro calls. for each macro call
  a) Locate macro name in MNT.
  b) Establish correspondance between formal parameters and actual parameters.
  c) Obtain information from MNT regarding position of macro definition in MDT.
  d) Expand macro call by picking-up statements from MDT.

* Conclusion:

- Hence, we have implemented pass - II of two-pass macroprocessor by taking input of assignment - 3

Author
39

* Title:
- Write a java program to implement following scheduling algorithm : FCFS, SJF (Pre-emptive), Priority (Non-Pre-emptive) and Round Robin (Pre-emptive)

* Objectives:
- To understand OS and scheduling concepts.
- To implement Scheduling FCFS, SJF, RR and Priority Algorithm
- To study about Scheduling and Scheduler.

* Problem Statement:
- Write a java program to implement scheduling algorithm FCFS, SJF, Priority and Round Robin.

* Outcomes:
- Students will be able to
1) Knowledge Scheduling policies.
2) Compare different scheduling algorithm.

* Software / Hardware Requirement:
- JDK / Eclipse.
- M/c Lenovo Think Center M700 Ci3, 6100, 6th Gen, H81, 4GB RAM, 500GB HDD

* Theory:
- CPU Scheduling:
- It refers to set of policies and mechanism built into the operating systems that govern the order in which the work to be done by a computer is completed.

Page No.
Date

- The main objective of scheduling is to optimize system performance in accordance with criteria deemed most important by the system designers

- what is scheduling ?
  - Scheduling is defined as process that governs the orders in which the work is to be done.
  - Scheduling is done in the areas where no. of jobs or works are to be performed.
  - CPU scheduling is best example of scheduling.

- what is scheduler ?
  - Scheduler in an OS module that selects the next job to be admitted into the system and the next process to run.
  - Primary objective of scheduler is to optimize system performance in accordance with the criteria deemed by system designer.

- Necessity of scheduling
  1) It is required when no. of jobs are to be performed by CPU
  2) Provides mechanism to give order to each ~~other~~ work to be done.
  3) Primary objective of scheduling is to optimize system performance.
  4) Scheduling provides ease to CPU to execute processes in efficient manner.

- Types of scheduler
  1) Long-term scheduler.
  2) Medium- term scheduler.
  3) Short-term scheduler.

Teacher's Sign.:

- Types of scheduling algorithm:
  1) FCFS (First Come First Serve)
  2) SJF (Short Job First)
  3) Priority Scheduling.
  4) Round Robin scheduling.

1) FCFS:
- It is working on simplest scheduling discipline.
- FCFS may result into poor performance.
- Advantages:
  a) Better for Long processes.
  b) No Starvation.
  c) Simple Method.
- Disadvantages:
  a) Throughput is not emphasized.

2) SJF:
- Best approach to minimize waiting time.
- The processor should know in advance how much time proce
  will take.
- Advantages
  a) Throughput is high.
- Disadvantages
  a) Starvation may be possible for longer processes.
- This algorithm is divided into 2 types:
  a) Pre-emptive SJF.
  b) Non-Preemptive SJF.

Teacher's Sign.: _____

3) Priority Scheduling:

- It is non-preemptive algorithm and one of the most common scheduling algorithm in batch systems.
- Each process is assigned a priority. Process with highest priority is to be executed first and so on.
- Advantages:
  a) Good responses for the highest priority processes.
- Disadvantages:
  a) Starvation may be possible for lowest priority processes.

4) Round-Robin Scheduling:

- Round-Robin is the preemptive process scheduling algorithm.
- Each process is provided a fix time to execute, it is called Quantum.
- Advantages:
  a) Fair treatmeant for all processes.
  b) Overhead on processor is low.
  c) Good response time for short processes.
- Disadvantages:
  a) Throughput is low. if quantum is too small.
  b) Care must be taken in choosing quantum value.

Page No.
Date

* Algorithms :

1) FCFS :

- Step 1 : Start the process.
Step 2 : Accept the no. of processes in ready queue.
Step 3 : Assign process id and accept CPU burst time.
Step 4 : Set waiting time of 1st process as '0' and its burst time as its turnaround time.
Step 5 : Calculate waiting time and Turn around time.

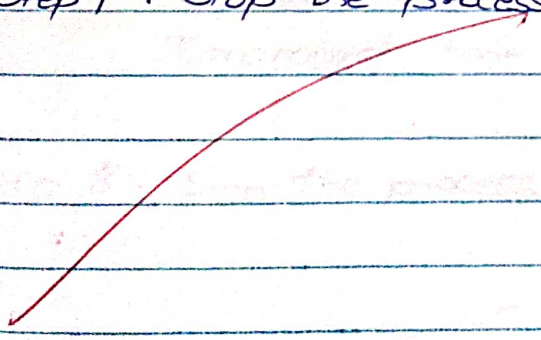
Waiting time = Waiting time of process (n-1) + Burst time of process (n-1)

Turnaround time = Waiting time of process (n) + Burst time for process (n)

Step 6 : Calculate

Average waiting time = Total Waiting Time / No. of proce
Average Turnaround time = Total Turnaround Time / No. of proces

Step 7 : Stop the process.



Page No.
Date

2) SJF

Step 1: Start the process

Step 2: Accept No. of process

Step 3: Assign process ID and accept CPU burst time.

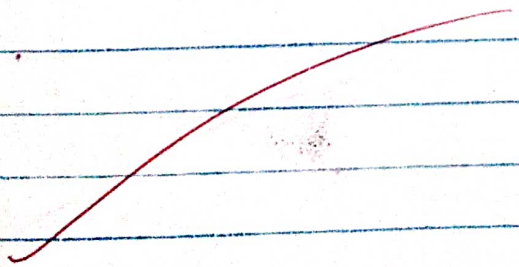
Step 4: Start Ready Q a/c to shortest Burst time

Step 5: Set waiting time of 1st process as '0' and turnaround time as its burst time.

Step 6: Calculate ~~Average~~ Waiting Time and ~~Average~~ Turnaround time

Step 7: Calculate ~~Gcait~~ Average waiting time and Average Turnarocurd time.

step 8: Stop the process.



Teacher's Sign.:

Page No.
Date

3) Round Robin

Step 1: Start the process.

Step 2: Accept No. of process in ready Q

Step 3: Assign process id and accept CPU burst time.

Step 4: Calculate No. of time slices for each process.
No. of time slice for process (n) = burst time process(n) / time slice.

Step 5: If burst time is less than time slice then
No. of time slice = 1.

Step 6: Calculate, waiting time for process (n), Turnaround time for process (n).

Step 7: Calculate, Average waiting time, Average Turnaround time.

Step 8: Stop the process.

Teacher's Sign.: _____

Page No.
Date

4) Priority Scheduling:

Step 1: Start the process.

Step 2: Accept No. of process.

Step 3: Assign process id and accept CPU burst time, Priority

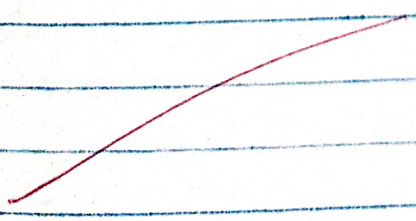
Step 4: Start Ready Q according to priority.

Step 5: Set waiting time of 1st process as '0' and its turnaround time as its burst time.

Step 6: ~~Set~~ Calculate waiting time for process (n), Turnaround time for process (n).

Step 7: Calculate Average Waiting time, Average turnaround time.

Step 8: Stop the process.



Teacher's Sign.: _____

Page No.
Date

* Conclusion :

- Hence, we have studied that
  1) Different CPU scheduling algorithms like FIFO, SJF, etc.
  2) CPU scheduling like context switching, types of scheduler, different timing parameters like Waiting Time, Turnaround time, burst time, etc.

Ashwini
34

Title:

- Page Replacement policies LRU and OPT

Objectives:

- To understand Page replacement policies.
- To understand paging concept.
- To understand page fault, page hit, miss, hit ratio, etc.

problem statement:

- Write a java program to implement page replacement policies LRU and OPT.

Outcomes:

- students will be able to
  1) Implement page replacement policies.
  2) Understand paging Concept.
  3) Understand page fault, page hit, etc.

S/w / H/w requirements:

- Latest JDK / Eclipse.
- M/c Lenovo Think Center, 4GB RAM, 500GB HDD.

Theory:

- Paging:
  - Paging is a memory management technique in which process address space is broken into blocks of same size called pages.
  - The size of process is measured in number of pages.

Similarly, main memory is divided into small-fixed blocks of (physical) memory called frames.
The size of a frame is kept the same as that of a page to have optimum utilization of main memory and to avoid external fragmentation.

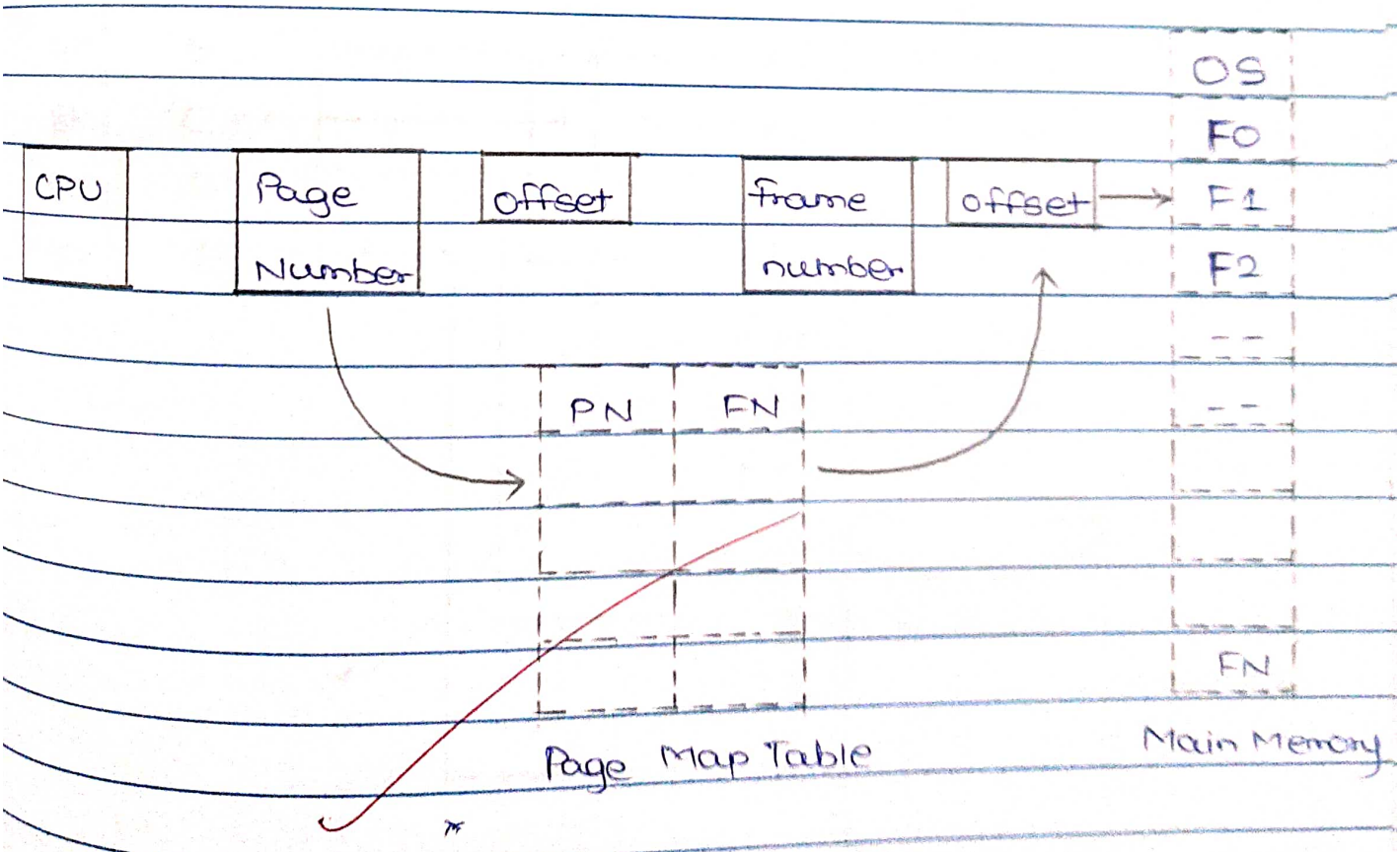
Address Translation:

- Page address is called logical address and represented by page number and the offset.

  Logical Address = Page Number + Page offset

- frame address is called physical address and represented by frame number and the offset.

  Physical Address = frame number + page offset.



Advantages of Paging and Disadvantage of Paging.

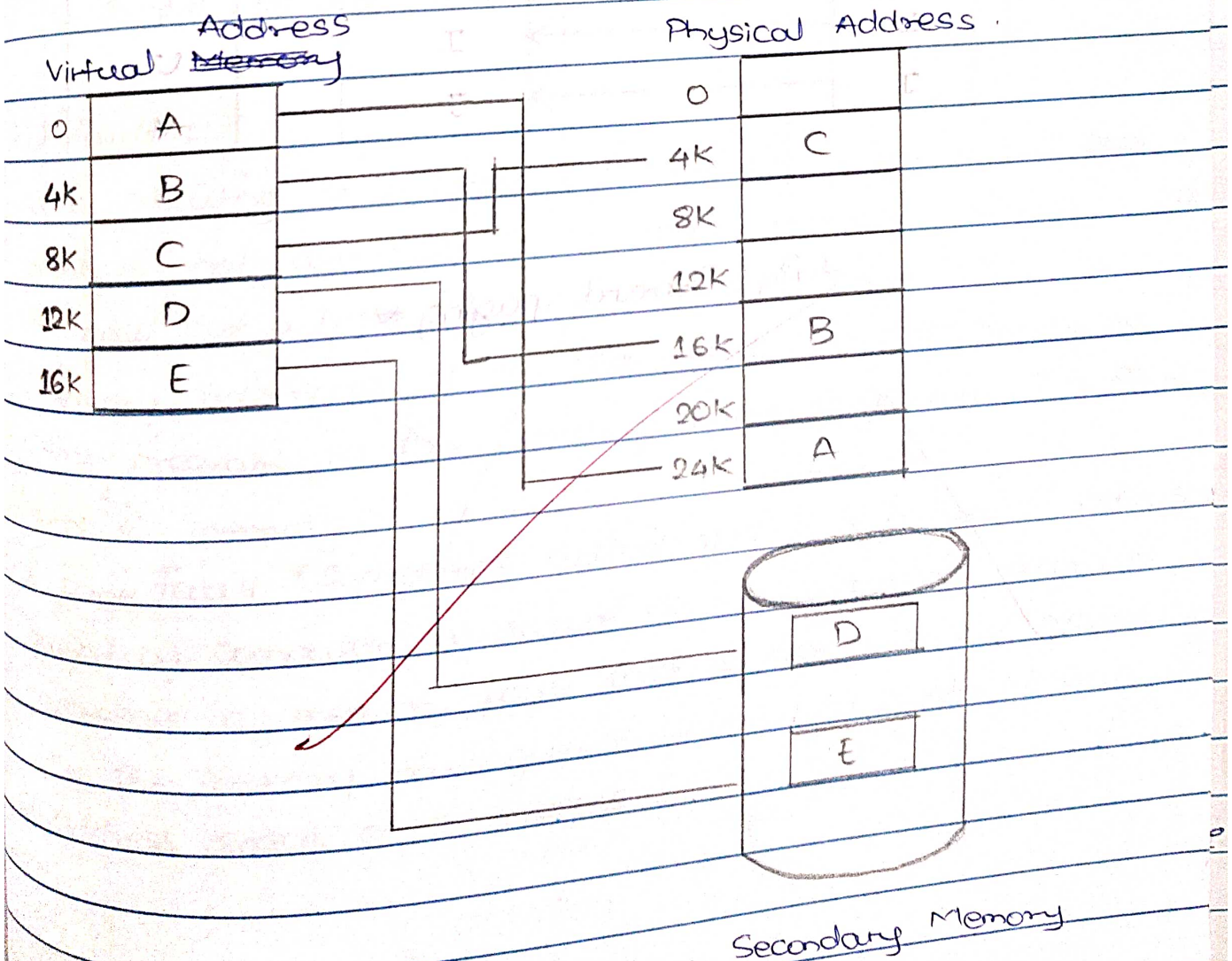
1) Reduces external fragmentation, but still suffers from internal fragmentation

2) Due to equal size of pages and frames, swapping becomes very easy.

3) Page table requires extra memory space, so may not be good for a system having small RAM.

A computer can address more memory than the amount physically installed on the system.

This extra memory is called actually a Virtual Memory.



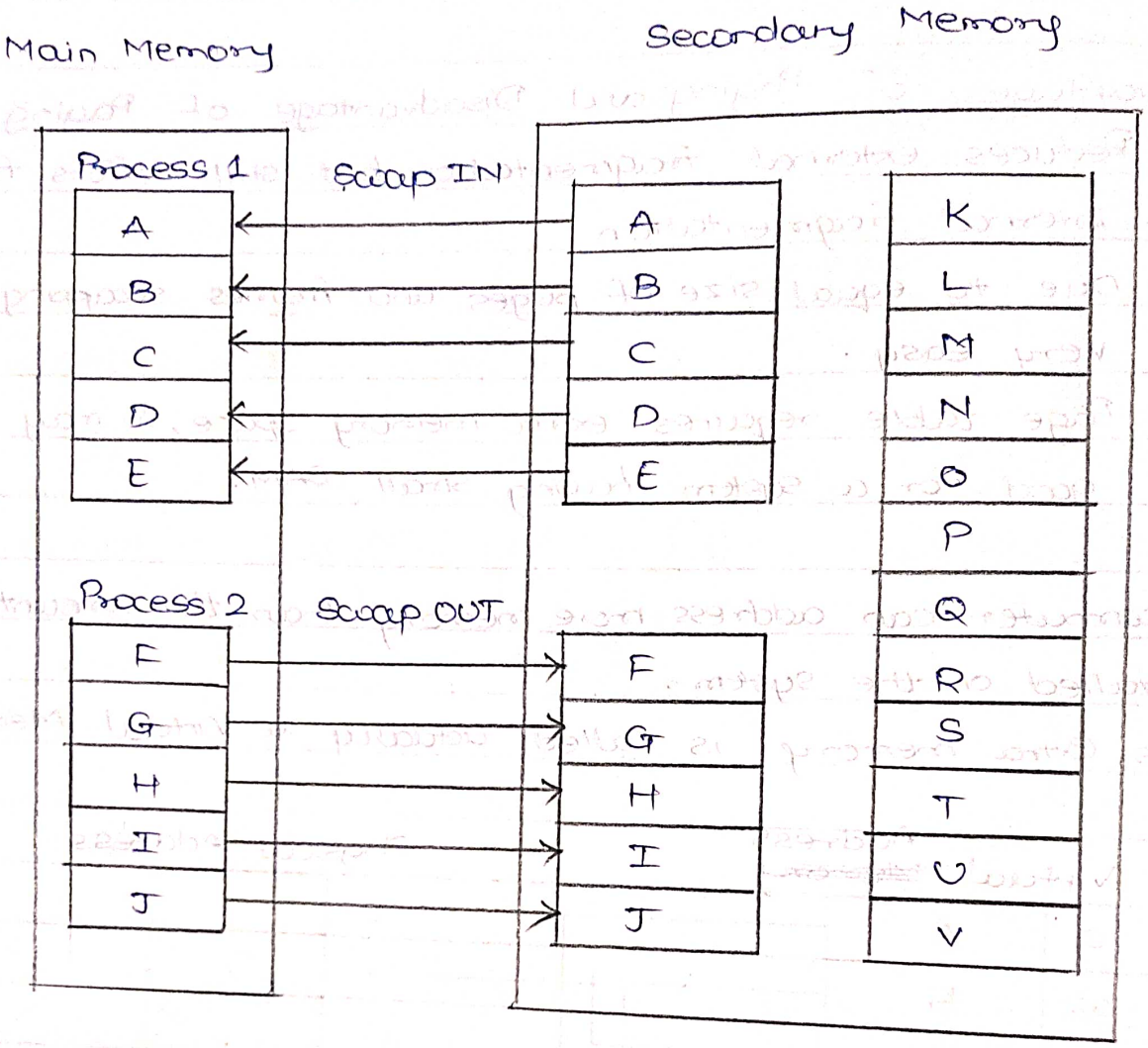


fig: Demand paging

- Demand Paging :

- A demand paging is quite similar to paging system with swapping when processes reside in secondary memory and pages are loaded only on demand, not in advance.

- Advantages :

1) Large Virtual Memory.

2) More efficient use of memory.

3) There is no limit on degree of multiprogramming.

- Disadvantages :

1) No. of tables and the amount of processor overhead for handling page interrupts are greater than in case of simple paged management techniques.

- Page fault :

- while executing a program, if program references a page which is not available in the main memory because it was swapped out a little ago, the processor treats this invalid memory references as a Page fault and transfers control from program to the operating system to demand the page back to memory.

- A page fault (Sometimes called #PF) is a type of exceptio raised by computer hardware when a running program accessess a memory page that is not currently mapped by the memory management Unit (MMU) into virtual address space of a process.

Page replacement Algorithm.

- Page replacement algorithm are the techniques using which an operating system decides which memory pages to scoap out, write to disk when a page cannot be used for allocation purpose accounting to reasons that pages are not available or the number of free pages is lower than required pages.
- A page replacement algorithm looks at limited information about accessing pages provided by h/w and tries to select which pages should be replaced ~~by~~ to minimize total no. of page misses, while balancing it with the costs of primary storage and processor time of the algorithm itself.

Page hit:

- A hit is a request to web server for a file like a web page, image, Javascript or css.
- When a web pages is downloaded from a server the no. of "hits" or "page hits" is equal to no. of files requested.

Page frame:

- The page frame is the storage unit whereas the page is contents that you could store in storage unit i.e page frame.
- RAM is divided into fixed size blocks called page frames which is typically 4KB in size. and each frame can store 4KB of data i.e the page.

Page table:

- A page table is the data structure used by virtual memory system in a computer operating system to store mapping between virtual addresses and physical addresses

Reference String:

- The string of memory reference is called reference string.
- Reference string are generated artificially or by tracing a given system and recording the addresses of each memory references.

FIFO Algorithm:

- Oldest page in main memory is the one which will be selected for replacement.
- Easy to implement, keep a list, replaces pages from tail and add new pages at head.

Optimal Page Algorithm:

- An optimal page-replacement algorithm has the lowest page-fault rate of all algorithms.
- An optimal page-replacement algorithm exists, and has been called OPT or MIN.

Least Recently Used (LRU) algorithm:

- Page which has not been used for the longest time in main memory is the one which will be selected for replacement.
- Easy to implement, keep a list, replace pages by looking back into time.

Conclusion :

- Hence, we have successful implemented page replacement policies LRU and OPT.

Rishikesh 39