

CS Bridge Module 25 Memory Management

1. Memory Management

1.1 CS Bridge: Memory Management



NYU TANDON
ONLINE

CS Bridge: Memory Management

Module 25
Dan Katz

Notes:

1.2 In this module

In This Module



- The need for memory management
- Memory Management requirements
- Logical vs. physical addresses
- Partitioning strategies
- Virtual memory
- Working Set strategies
- Resident Set Strategies
- Load control
- Shared pages



1.3 Reasons for Memory Management

Reasons for Memory Management

NYU TANDON
ONLINE

- In a multiprogramming system, there will never be enough main memory
- The OS will need to allocate memory to multiple programs running in the system
- The OS will need to move parts between main memory and secondary memory



1.4 Memory Management Requirements

Memory Management Requirements

NYU TANDON
ONLINE

- Relocation
- Protection
- Sharing
- Local Organization
- Physical Organization

Hover each topic for more information.



1.5 Logical vs. Physical addresses

Logical vs. Physical Addresses

NYU TANDON
ONLINE

- Due, primarily, to relocation, a program will have to use a logical address to access memory
- Logical addresses need to be converted, dynamically, to physical addresses
- A simple solution to logical addresses could be the offset from the beginning of the program
- The CPU's Hardware Memory Management unit is responsible for converting, during runtime, the logical address to a physical address



1.6 Partitioning Strategies

Partitioning Strategies

NYU TANDON
ONLINE

- Fixed Partitioning
- Dynamic Partitioning
- Buddy System
- Simple Paging
- Simple Segmentation



1.7 Fixed Partitioning

Fixed Partitioning

NYU TANDON
ONLINE

- Main memory divided into a fixed number of partitions
- A process is allocated a part of main memory equal or less than its size
- Equal Sized Partitioning
 - Causes fragmentation and wasted space in partition
- Unequal Sized Partitioning
 - Need to worry about which partition to place the process in



Equal Sized Partitioning	Unequal Sized Partitioning
8	8
8	12
8	16
8	8
8	14
2	2

1.8 Dynamic Partitioning

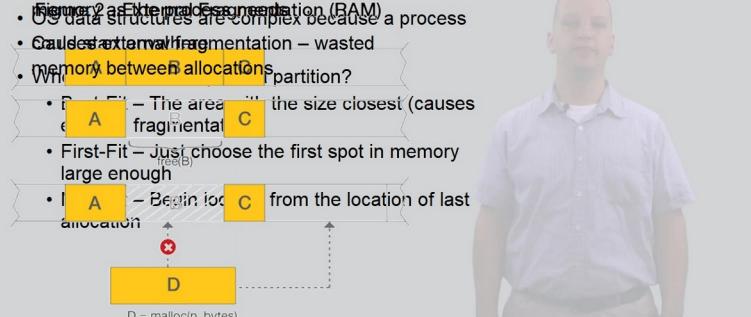
Dynamic Partitioning

NYU TANDON
ONLINE

- Reduces contention, forces defragmentation
- Slower than fixed partitioning (RAM)
- OS data structures are complex because a process can move between partitions
- Causes external fragmentation – wasted memory between allocations

What is the best fit partition?

- Best Fit – The area with the size closest (causes external fragmentation)
- First-Fit – Just choose the first spot in memory large enough
- Last Fit – Begin location from the location of last allocation



D = malloc(n_bytes)



1.9 Buddy System

Buddy System

NYU TANDON
ONLINE

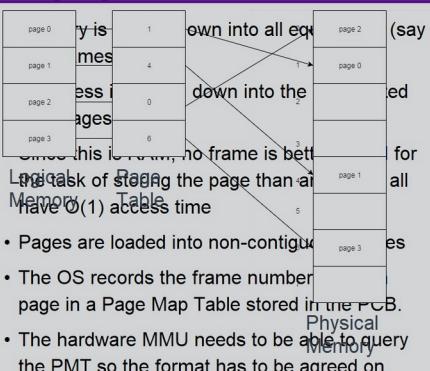
- A compromise between fixed and dynamic
- Memory is divided and divided again in multiples of 2
- 2MB of memory might be divided as such, if we need a 100 Kb allocation
- The process would be given the first 128 Kb space
- If we have a large number of 100 Kb processes we would further divide the larger partitions to accommodate the need.
- This makes the OS structures easier because processes begin and end on a 2^n boundary



1.10 Paging

Paging

NYU TANDON
ONLINE



1.11 Benefits of Paging

Benefits of Paging

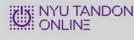
NYU TANDON
ONLINE

- No external fragmentation
- A minimum of internal fragmentation
- Easy protection
- Easy relocation
- Easy sharing



1.12 Converting Logical to Physical

Converting Logical to Physical



- The logical address is relative to the start of the process
- To calculate the PAGE number, we do bit shifts because pages are always multiples of 2
- To calculate the OFFSET into the page, we can use an XOR
- The page number can be used as an index (like in an array) into the PMT to find the Frame number in main memory
- The CPU will then bit shift that frame number and add the offset to find the physical address
- $\text{physicalAddr} = \text{lookupInPMT}(\text{logicalAddr}/\text{pageSize}) * \text{pageSize} + \text{logicalAddr} \% \text{pageSize}$



1.13 Segmentation

Segmentation



- Each process is divided into unequal sized partitions
- Logical Address now needs to be Segment# + Offset
- Storage is similar to Dynamic Partitioning



1.14 Knowledge Check

(Multiple Response, 10 points, 5 attempts permitted)

Knowledge Check

List some Partitioning Strategies (select all that apply).

- Fixed/Dynamic Partitioning
- Buddy System
- Simple Paging
- Simple Segmentation
- Relocation
- Sharing
- Logical Addressing
- Physical Addressing

Correct	Choice
X	Fixed/Dynamic Partitioning
X	Buddy System
X	Simple Paging
X	Simple Segmentation
	Relocation
	Sharing
	Logical Addressing
	Physical Addressing

Feedback when correct:

That's right! You selected the correct response.

Feedback when incorrect:

Are these concepts for partitioning or implementation strategies?

Correct (Slide Layer)

Knowledge Check

NYU TANDON
ONLINE

List some Partitioning Strategies (select all that apply).

Fixed/Dynamic Partitioning

Buddy System

Simple Partitioning

Simple Segmentation

Relocation

Sharing

Logical Addressing

Physical Addressing

Correct

That's right! You selected the correct response.

Continue

Incorrect (Slide Layer)

Knowledge Check

NYU TANDON
ONLINE

List some Partitioning Strategies (select all that apply).

Fixed/Dynamic Partitioning

Buddy System

Simple Partitioning

Simple Segmentation

Relocation

Sharing

Logical Addressing

Physical Addressing

Incorrect

Are these concepts for partitioning or implementation strategies?

Continue

Try Again (Slide Layer)

Knowledge Check

NYU TANDON
ONLINE

List some Partitioning Strategies (select all that apply).

Fixed/Dynamic Partitioning

Buddy System

Simple Partitioning

Simple Segmentation

Relocation

Sharing

Logical Addressing

Physical Addressing

Incorrect

That is incorrect. Please try again.

Try Again

1.15 Virtual Memory

Virtual Memory

NYU TANDON
ONLINE

- ~~uses Paging or Segmentation~~
- ~~the process will only ever need memory location at a given time, so the memory of the process doesn't have to be in MAIN memory.~~
- ~~can swap pages of the program from hard drive and bring them back later, if needed~~
- Example: The splash screen at the beginning of a process isn't ever needed again once the process is running



The slide contains several crossed-out text snippets, likely from a previous version of the content. These include:

- uses Paging or Segmentation
- the process will only ever need memory location at a given time, so the memory of the process doesn't have to be in MAIN memory.
- can swap pages of the program from hard drive and bring them back later, if needed

1.16 Knowledge Check

(Multiple Response, 10 points, 4 attempts permitted)

Knowledge Check

NYU TANDON
ONLINE

What partitioning strategy can be used for Virtual Memory? (Select all that apply)

Fixed/Dynamic Partitioning

Buddy System

Simple Paging

Simple Segmentation

Correct	Choice
	Fixed/Dynamic Partitioning
	Buddy System
X	Simple Paging
X	Simple Segmentation

Feedback when correct:

That's right! You selected the correct response.

Feedback when incorrect:

You did not select the correct response.

Correct (Slide Layer)

Knowledge Check

NYU TANDON
ONLINE

What partitioning strategy can be used for Virtual Memory? (Select all that apply)

Fixed/Dynamic Partitioning

Buddy System

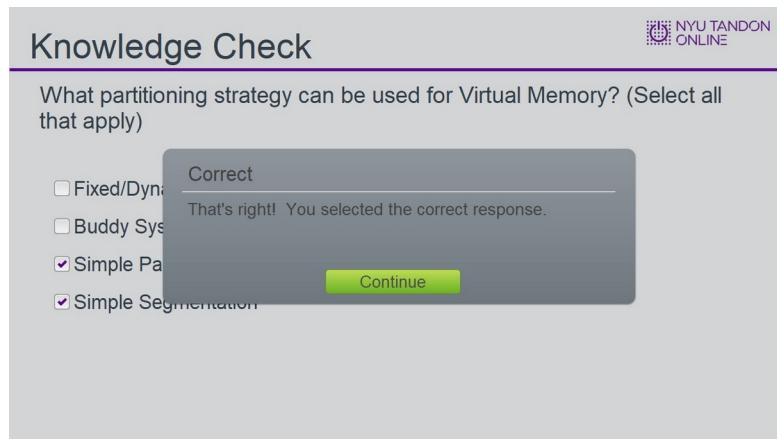
Simple Paging

Simple Segmentation

Correct

That's right! You selected the correct response.

Continue



Incorrect (Slide Layer)

Knowledge Check

NYU TANDON
ONLINE

What partitioning strategy can be used for Virtual Memory? (Select all that apply)

Fixed/Dynamic Partitioning

Buddy System

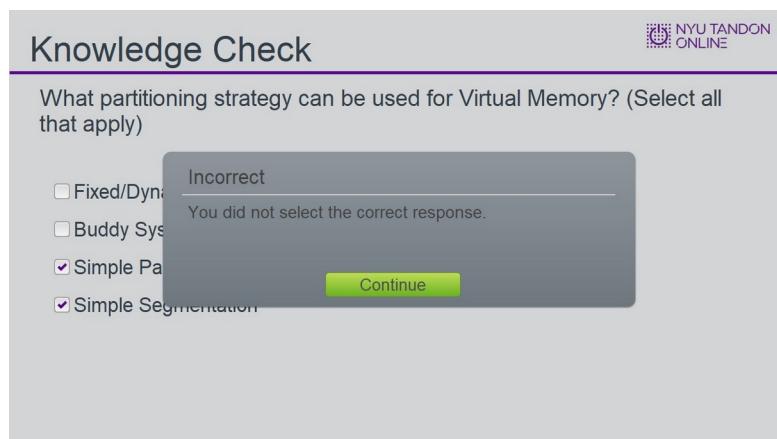
Simple Paging

Simple Segmentation

Incorrect

You did not select the correct response.

Continue



Try Again (Slide Layer)

Knowledge Check

NYU TANDON
ONLINE

What partitioning strategy can be used for Virtual Memory? (Select all that apply)

Fixed/Dyn: Incorrect
 Buddy Sys
 Simple Pa
 Simple Segmentation

That is incorrect. Please try again.

Try Again

1.17 What stays?

What Stays?

NYU TANDON
ONLINE

Resident Set Working Set

PMT



PMT (Slide Layer)

What Stays?

NYU TANDON
ONLINE

Resident Set Working Set

PMT

- The PMT contains a "Present" bit to indicate what pages are present in main memory
- The PMT contain a "Modify" bit to indicate if the copy of the page on the hard drive is the same as the copy in memory.
- If the MMU encounters a page which the process wants but is NOT in the resident set, it is called a "Page fault" and the CPU switches to running the OS (like an interrupt)



Working Set (Slide Layer)

What Stays?

NYU TANDON
ONLINE

Resident Set Working Set

PMT

- Working Set – That portion of the process which is in use
- In order for the process to run, the working set must be in the resident set.



Resident Set (Slide Layer)

What Stays?

NYU TANDON
ONLINE

Resident Set Working Set

PMT

- Resident Set – That portion of the process in main memory



1.18 Benefits of VM

Benefits of VM

NYU TANDON
ONLINE

- The programmer perceives a much larger memory space
- The system can remove unused pages
- More processes can run in the system resulting in better performance



1.19 Lookup problems

Lookup Problems

NYU TANDON
ONLINE

The diagram shows the process of translating a virtual address into a physical address. It starts with a **Virtual Address** (Page # and Offset) which is first tried in a **TLB**. If a **TLB miss** occurs, it uses a **PTE** (Physical Address and Physical Page #) from the PMT to access **Physical Memory**. The memory is organized into pages (0-6). A callout box highlights that each memory access requires two memory accesses: one through the TLB and one through the PMT.

- Not every memory access by the process requires that the MMU determine the physical address, which requires a query to the PMT
- Each memory access requires two memory accesses
- To save time, the CPU implements a Translation Lookaside Buffer which is a type of fast, addressable memory which stores a cache of those entries in the PMT which have been retrieved recently.

Notes:

1.20 Replacement policy

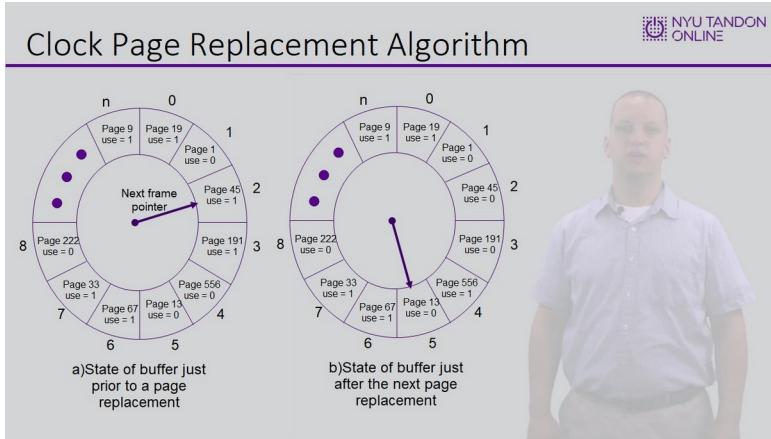
Replacement Policy

NYU TANDON
ONLINE

The diagram lists several points about replacement policies:

- When main memory runs out, the OS needs to remove some pages, this is called stealing
- If we choose poorly, the result will be worse performance
- The optimal choice would be to steal the page which is not going to be used for the longest period of time
- Possible Algorithms
 - Optimal – Not feasible
 - LRU – Not feasible
 - FIFO – Easy to implement but poor performance
 - Clock – Easy to implement but requires USE bits

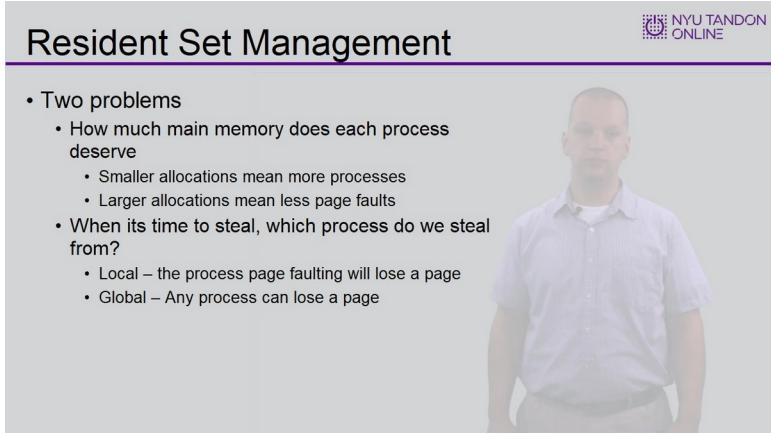
1.21 Clock page replacement algorithm



Notes:

Image from <http://image.slidesharecdn.com/os-pagereplacementalgorithms-150104051823-conversion-gate02/95/os-page-replacement-algorithms-26-638.jpg?cb=1420370365>

1.22 Resident set management

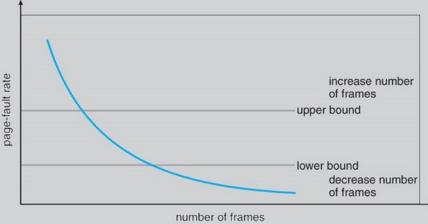


1.23 Controlling page faults by resident set size

Controlling Page Faults By Resident Set Size

NYU TANDON
ONLINE

- If a process fits entirely in main memory, it will not generate any page faults
- If a process only has one frame, it will generate the maximum number of page faults
- There is a non linear progression between those two points



Notes:

Image taken from:

https://www.cs.uic.edu/~jbell/CourseNotes/OperatingSystems/images/Chapter9/9_21_PageFaultFrequency.jpg

1.24 PFF Algorithm

PFF Algorithm

NYU TANDON
ONLINE

- Examine the time between the last page fault and the current one
 - If less than an preset F value, add a frame
 - If greater than the preset F value, discard all pages with use bit 0, reset all use bits to 0 and continue
- Upper bounds and lower bounds would be better than just F
- Unfortunately, it performs poorly during locality shifts



1.25 VSWS Algorithm

VSWS Algorithm

NYU TANDON
ONLINE

- VSWS Solves PFF's problems by setting
 - a minimum value for clearing unused pages (M)
 - A minimum number of page faults which must have occurred before clearing unused pages (Q)
 - A maximum duration before clearing unused pages (L)
- Below M time units, always add a frame
- Between M and L
 - If there have been less than Q page faults, add a frame
 - If there have been more than Q page fault, discard unused pages, reset use bits, and reset the timer and Q counter
- At L, discard unused pages, reset use bits, and reset the timer and Q counter



1.26 Load Control

Load Control

NYU TANDON
ONLINE

Lowest Priority	Hover for more information
Faulting Process	
Last Process Activated	
Smallest Local Resident Set	
Largest Process	
Largest Remaining Execution Window	



Faulting Process (Slide Layer)

Load Control

NYU TANDON
ONLINE

Lowest Priority	Hover for more information
Faulting Process	Greater probability that it does not have working set resident
Last Process Activated	
Smallest Local Resident Set	
Largest Process	
Largest Remaining Execution Window	



Last Process Activated (Slide Layer)

Load Control

NYU TANDON
ONLINE

Lowest Priority

Faulting Process

Last Process Activated

Smallest Local Resident Set

Largest Process

Largest Remaining Execution Window

Hover for more information

Least likely to have working resident set

Smallest Local Resident Set (Slide Layer)

Load Control

NYU TANDON
ONLINE

Lowest Priority

Faulting Process

Last Process Activated

Smallest Local Resident Set

Largest Process

Largest Remaining Execution Window

Hover for more information

Requires the least effort to reload at a later time.

Largest Process (Slide Layer)

Load Control

NYU TANDON
ONLINE

Lowest Priority

Faulting Process

Last Process Activated

Smallest Local Resident Set

Largest Process

Largest Remaining Execution Window

Hover for more information

Most Bang for your Buck

Largest Remaining Execution Window (Slide Layer)

The slide has a header 'Load Control' and a 'NYU TANDON ONLINE' logo. On the left is a vertical list of process types in purple boxes:

- Lowest Priority
- Faulting Process
- Last Process Activated
- Smallest Local Resident Set
- Largest Process
- Largest Remaining Execution Window

A placeholder image of a person is shown with the text 'Hover for more information' above it. A callout box points to the 'Largest Remaining Execution Window' item with the text 'Has the longest processing time'.

1.27 Knowledge Check

(Multiple Response, 10 points, 5 attempts permitted)

The slide has a header 'Knowledge Check' and a 'NYU TANDON ONLINE' logo. The question is: 'If there are too many page faults occurring, select the criteria for removing a process (Select all that apply).'

The list of options includes:

- Medium Priority
- Largest Process
- Process with the smallest remaining execution window
- Process with the smallest local resident set
- Faulting Process

Correct	Choice
	Medium Priority
X	Largest Process
	Process with the smallest remaining execution window
X	Process with the smallest local resident set
X	Faulting Process

Feedback when correct:

That's right! You selected the correct response.

Feedback when incorrect:

You did not select the correct response.

Correct (Slide Layer)

Knowledge Check

NYU TANDON
ONLINE

If there are too many page faults occurring, select the criteria for removing a process (Select all that apply).

Medium P **Correct**
 Largest Pr
 Process w
 Process With the Smallest Local Resident Set
 Faulting Process

That's right! You selected the correct response.

Continue

Incorrect (Slide Layer)

Knowledge Check

NYU TANDON
ONLINE

If there are too many page faults occurring, select the criteria for removing a process (Select all that apply).

Medium P **Incorrect**
 Largest Pr
 Process w
 Process With the Smallest Local Resident Set
 Faulting Process

You did not select the correct response.

Continue

Try Again (Slide Layer)

Knowledge Check

If there are too many page faults occurring, select the criteria for removing a process (Select all that apply).

Medium P
 Largest Pr
 Process w
 Process With the Smallest Local Resident Set
 Faulting Process

Incorrect
That is incorrect. Please try again.

Try Again

1.28 Shared Pages

Shared Pages

Using virtual memory and/or paging, we can understand how sharing of structures can occur, both processes have entries in their page table for the same frames

- Code – Since code doesn't change, sharing of code pages is easy
- Data – Since code changes frequently we can adapt a Copy On Write scenario

Copy On Write – Allows sharing of pages

- The page table is expanded to allow a "read only" attribute.
- If a page is marked RO, any attempt to change that page will result in a trap to the OS
- The OS can then copy the page and mark both the original and the copy as RW since its no longer shared

1.29 Unix FORK function

Unix FORK Function

ful in systems which don't have thread support, UNIX fork function creates an exact duplicate copy of the current process.

This is most easily done by creating a new process, and of course a new page table, but duplicating the data in the original page table.

Both tables, in their entirety, are marked read only.

There's no need to actually COPY the memory until something changes, and then only the page changing will be copied.

1.30 In this module we learned

What We've Learned

NYU TANDON
ONLINE

- The need for memory management
- Memory Management requirements
- Partitioning Strategies
- Logical vs. physical addresses
- Virtual memory
- Working Set strategies
- Resident Set Strategies
- Load control
- Shared pages



1.31 Results Slide

(Results Slide, 0 points, 1 attempt permitted)

Results

NYU TANDON
ONLINE

Your Score: %Results.ScorePercent%% (%Results.ScorePoints% points)

Passing Score: %Results.PassPercent%% (%Results.PassPoints% points)

Result:

[Retry Quiz](#)

[Review Quiz](#)

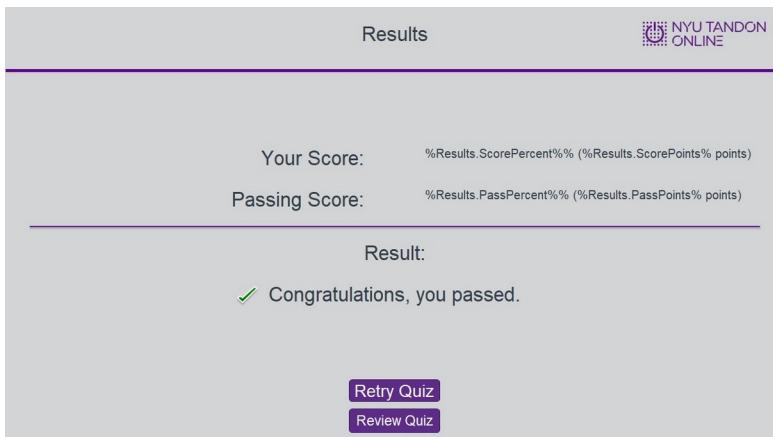
Results for
1.14 Knowledge Check
1.16 Knowledge Check
1.27 Knowledge Check

Result slide properties

Passing 80%

Score

Success (Slide Layer)



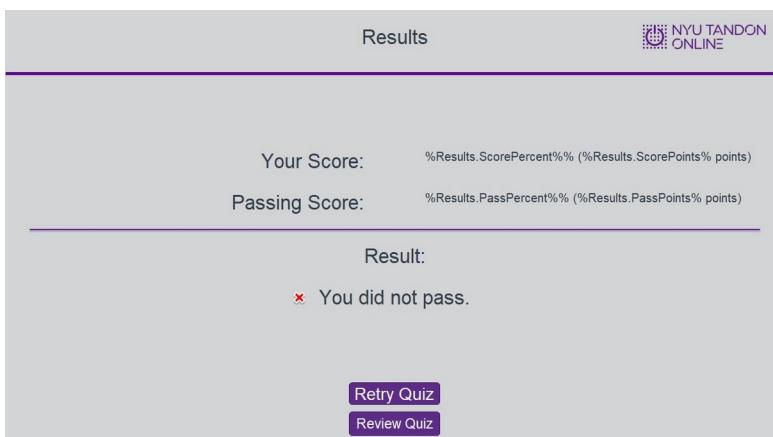
A screenshot of a slide titled "Results" from NYU Tandon Online. The slide displays the user's score and passing score, followed by a success message and two action buttons.

Your Score: %Results.ScorePercent%% (%Results.ScorePoints% points)
Passing Score: %Results.PassPercent%% (%Results.PassPoints% points)

Result:
✓ Congratulations, you passed.

[Retry Quiz](#)
[Review Quiz](#)

Failure (Slide Layer)



A screenshot of a slide titled "Results" from NYU Tandon Online. The slide displays the user's score and passing score, followed by a failure message and two action buttons.

Your Score: %Results.ScorePercent%% (%Results.ScorePoints% points)
Passing Score: %Results.PassPercent%% (%Results.PassPoints% points)

Result:
✗ You did not pass.

[Retry Quiz](#)
[Review Quiz](#)

1.32 End of Module



End of Module

Exit