CSCI315 – Operating Systems Design

Department of Computer Science
Bucknell University

Introduction to Memory Management
Memory Labs Overview

Ch 9.1-9.2

This set of notes is based on notes from the textbook authors, as well as L. Felipe Perrone, Joshua Stough, and other instructors.
Xiannong Meng, Fall 2021.

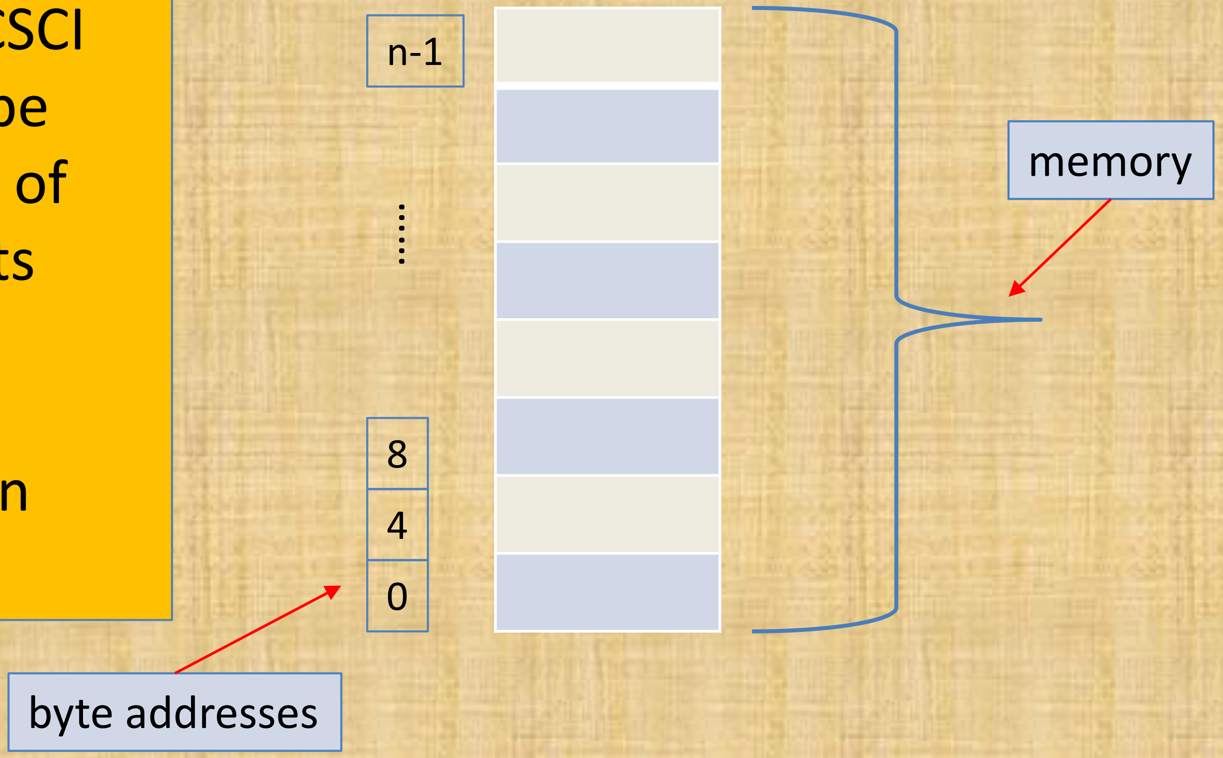
Background

- Source programs such as a C program must be compiled and linked with library to form an executable.
- An executable program must be brought from disk into memory and placed within a process for it to be run.
- ***Waiting queue*** – collection of processes on the disk that are waiting to be brought into memory to run the program (long term scheduler).

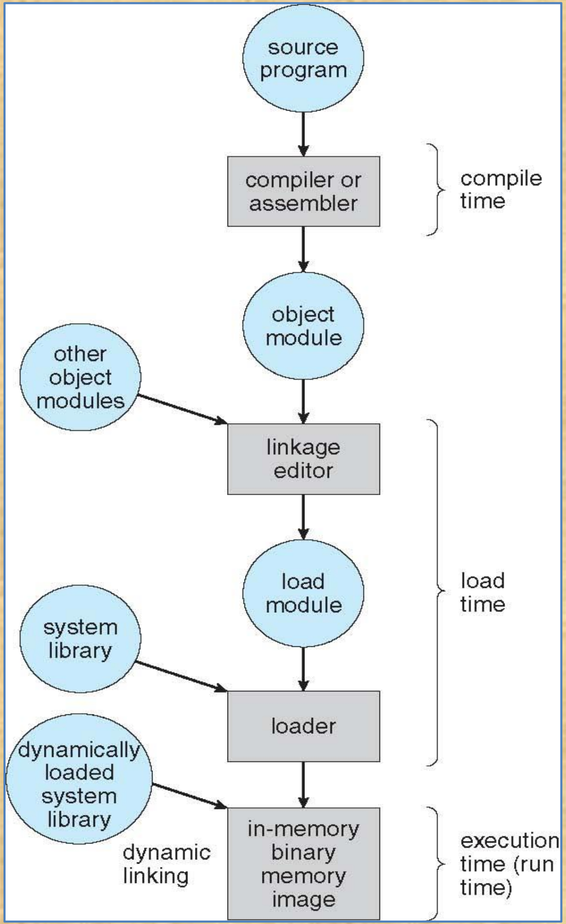
Logical View of Memory

We learned from CSCI 206, memory can be viewed as an array of words, each with its address.

Addresses can be in bytes or words.



Processing of a User Program




```
gcc -c hello.c -o hello.o
as hello.s -o hello.o
```

```
ld -o hello hello.o /usr/lib64/crt1.o /usr/lib64/crti.o
```

Binding of Instructions and Data to Memory

Address binding of instructions and data to memory addresses can happen at three different stages:

- **Compile time**: If memory location known a priori, absolute code can be generated; must recompile code if starting location changes.
- **Load time**: Must generate *relocatable* code if memory location is not known at compile time.
- **Execution time**: Binding delayed until run time if the process can be moved during its execution from one memory segment to another. Need hardware support for address maps (e.g., *base* and *limit registers*). (Most modern OSes use a variation of this scheme.)

Logical vs. Physical Address Space

- The concept of a ***logical address space*** that is bound to a separate ***physical address space*** is central to proper memory management.

  - **Logical address** – generated by the CPU; also referred to as *virtual address*.
  - **Physical address** – address seen by the memory unit.

- Logical addresses must be mapped onto physical addresses when the program is loaded into memory for execution.

Memory-Management Unit (MMU)

- Hardware device that maps virtual to physical address.

- In MMU scheme, the value in the relocation register is added to every address generated by a user process at the time it is sent to memory.

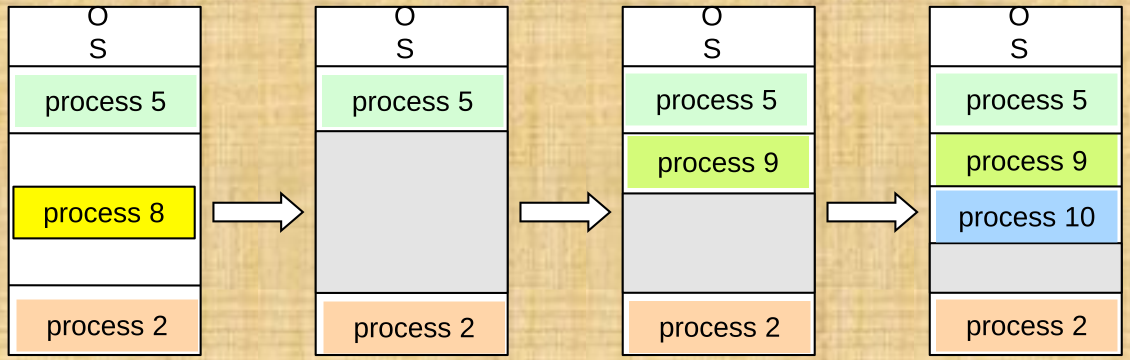
- The user program deals with *logical* addresses; it never sees the *real* physical addresses.

Contiguous Allocation

- Main memory usually is in two partitions:
  - Resident operating system, usually held in low memory with interrupt vector.
  - User processes then held in high memory.
- Single-partition allocation
  - Relocation-register scheme used to protect user processes from each other, and from changing operating-system code and data.
  - Relocation-register contains value of smallest physical address; limit register contains range of logical addresses – each logical address must be less than the limit register.

Contiguous Allocation

- Multiple-partition allocation
  - *Hole* – block of available memory; holes of various size are scattered throughout memory.
  - When a process arrives, it is allocated memory from a hole large enough to accommodate it.
  - Operating system maintains information about:
    a) allocated partitions   b) free partitions (hole)



Dynamic Storage-Allocation Problem

How to satisfy a request of size *n* from a list of free holes.

- **First-fit**: Allocate the *first* hole that is big enough.
- **Best-fit**: Allocate the *smallest* hole that is big enough; must search entire list, unless ordered by size. Produces the smallest leftover hole.
- **Worst-fit**: Allocate the *largest* hole; must also search entire list. Produces the largest leftover hole.

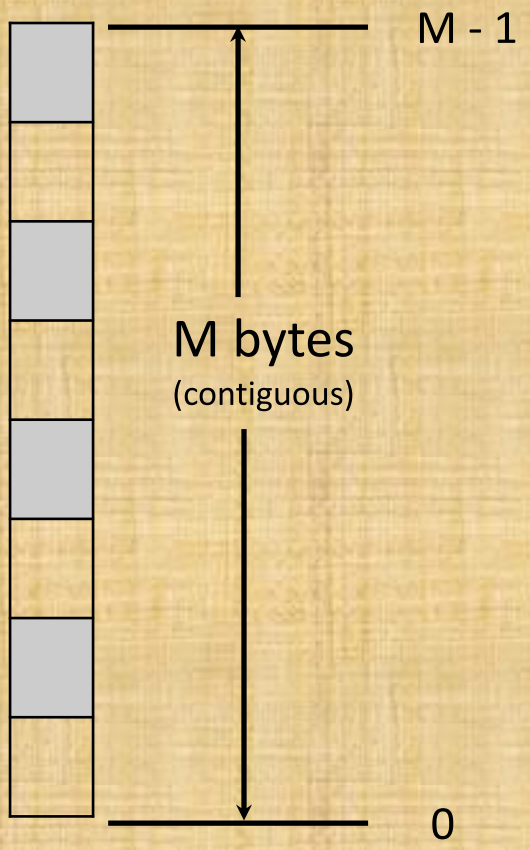
First-fit and best-fit better than worst-fit in terms of speed and storage utilization.

Sidebar:

The Memory Labs

A Custom Memory Allocator

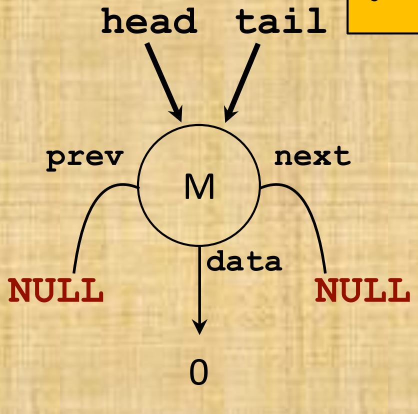
```
void *allocate (int size);

void deallocate (void *p);
```



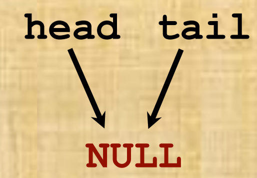
- the M bytes are obtained through **malloc()**
- **M** is the size of the initial free block
- **data** points to the first available address

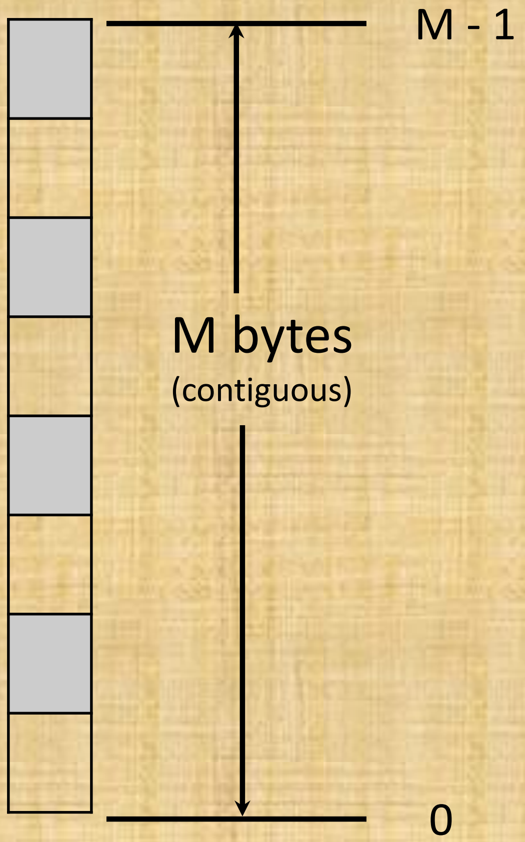
Initial State

free list

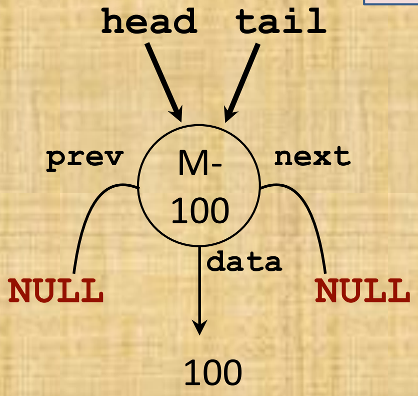


allocated list



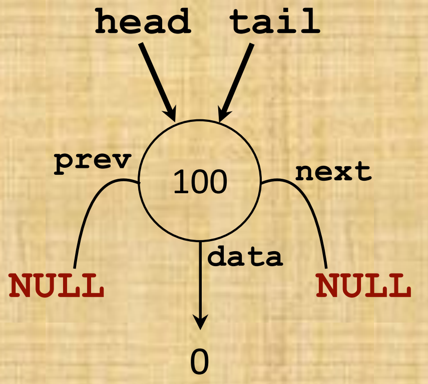


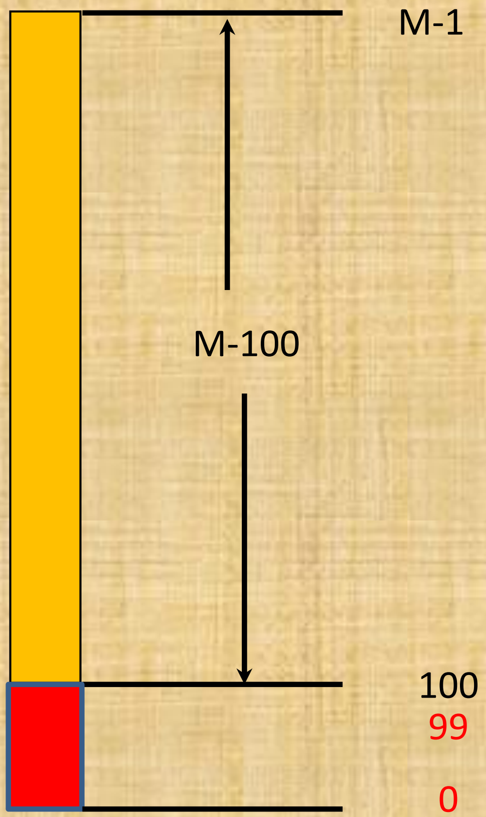
free




```
ptr1 = allocate(100);
```

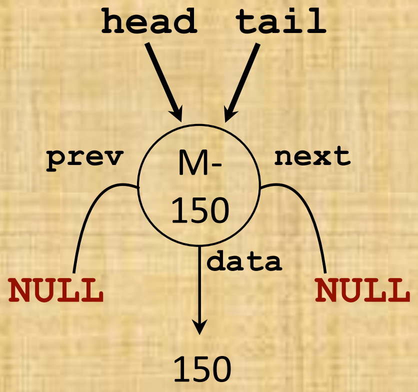
allocated



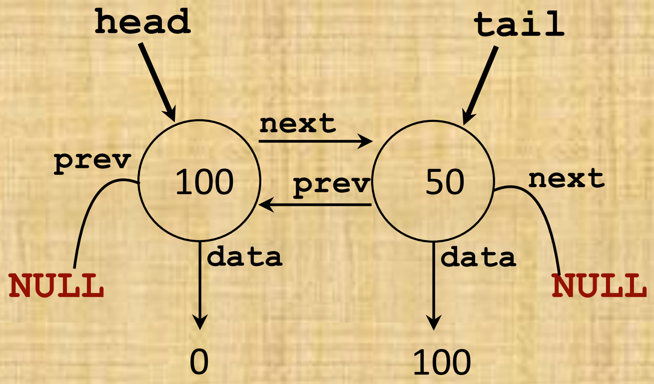


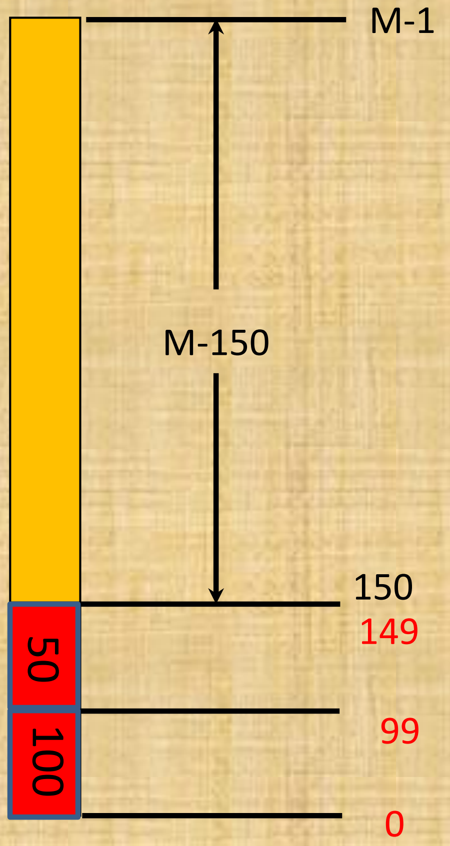
```
ptr2 = allocate(50);
```

free

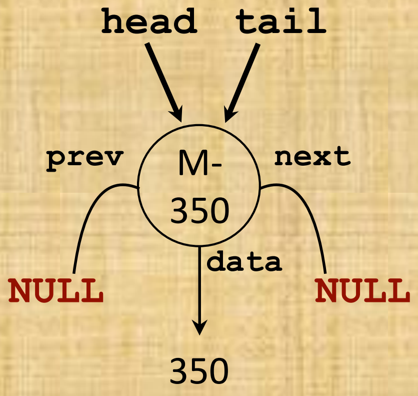


allocated



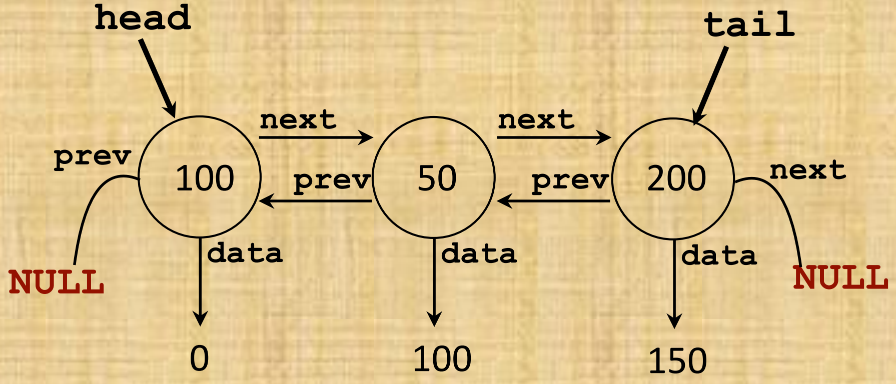


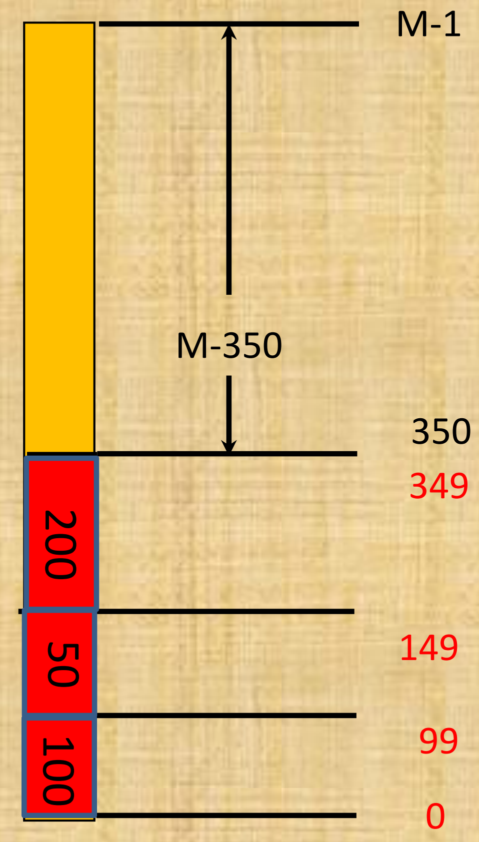
free



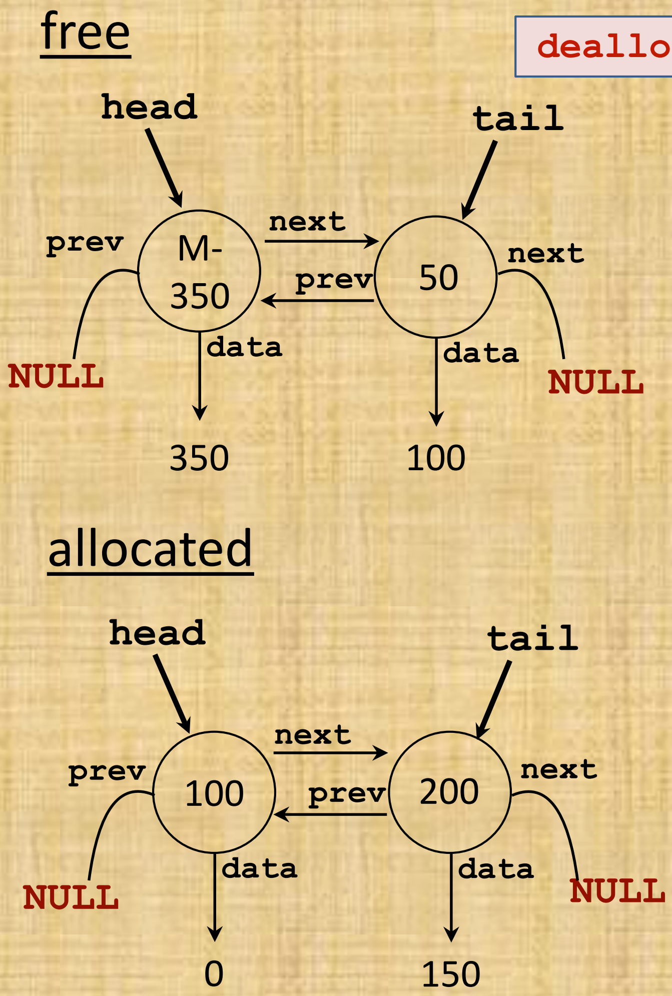

```
ptr3 = allocate(200);
```

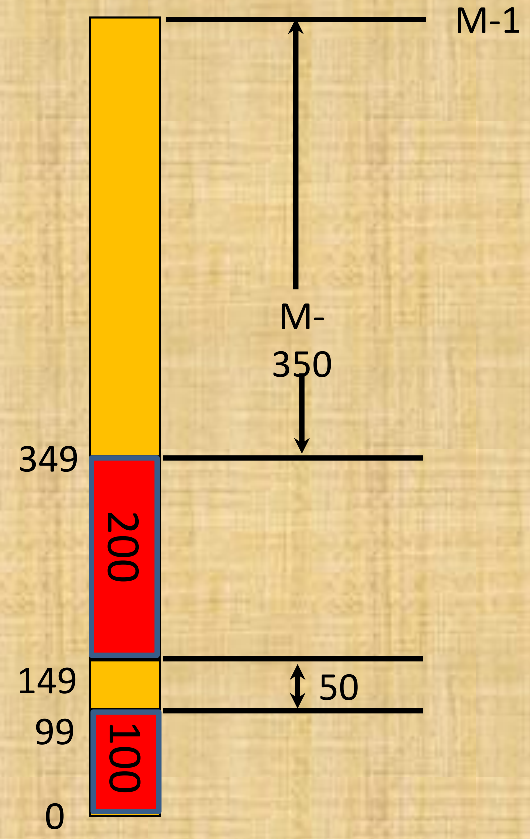
allocated





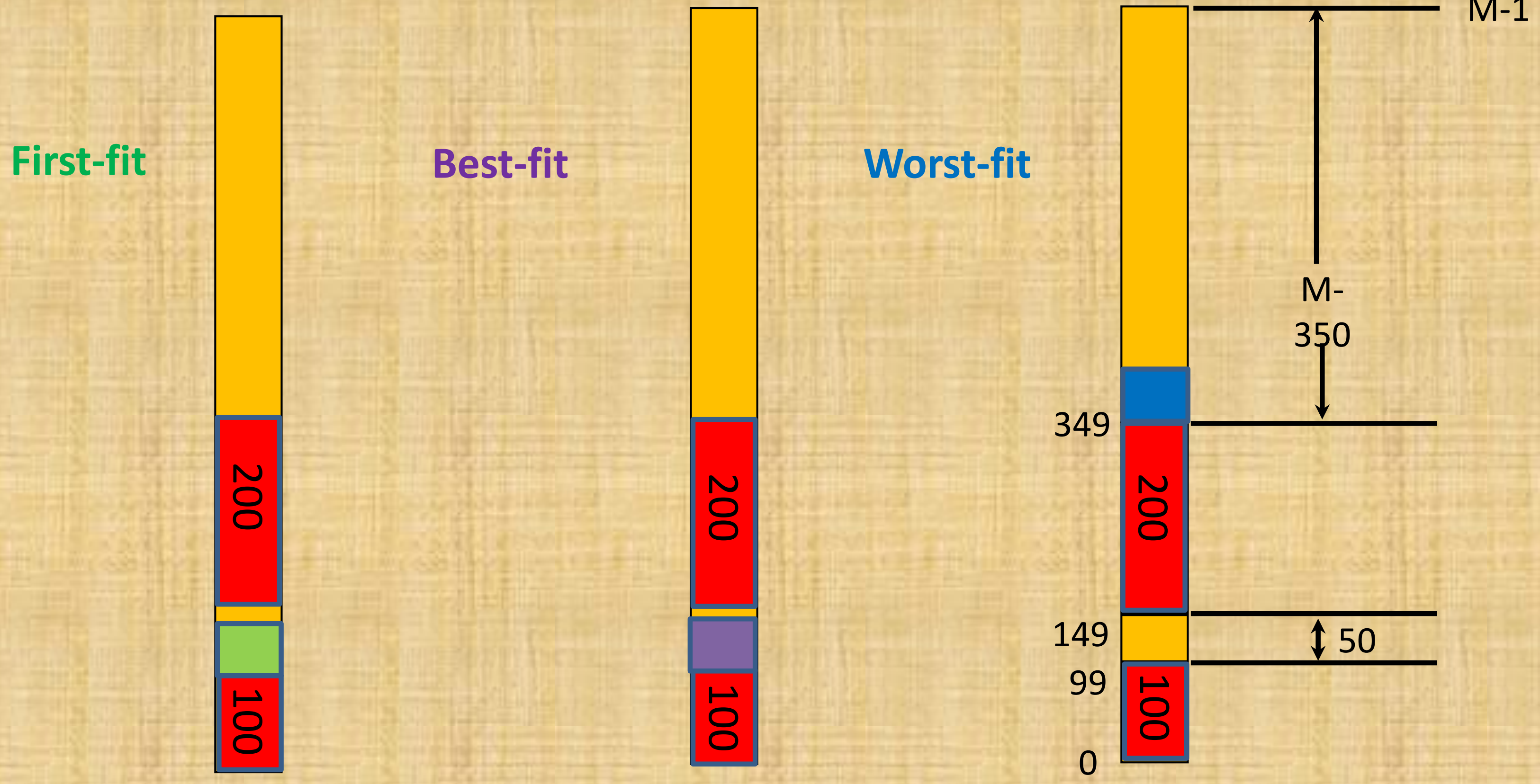
`deallocate(ptr2);`





```
ptr4 = allocate(40);
```

In this example, first-fit and best-fit result the same.



End of Sidebar