Miscellaneous Memory Management topics
Preparing Program for Execution

- Program Transformations
  - Translation (Compilation)
  - Linking
  - Loading
Address Binding

- Assign Physical Addresses = Relocation
- Static binding
  - Programming time
  - Compilation time
  - Linking time
  - Loading time
- Dynamic binding
  - Execution time
Static Address Binding

Static Binding = At Programming, Compilation, Linking, and/or Loading Time

source module

int i

f()

Compiler

object module

store 20

branch f

store module (in secondary memory)

Linker

load module (in main memory)

Loader

function f

store 120

branch 0

store 1120

branch 1000

function f

0

20

100

120

1000

1100

1120
Dynamic Address Binding

Dynamic Binding = At Execution Time

```
int i
i = ... f()
```

```
store 20
branch f
```

```
store 120 branch 0
```

```
store 120 branch 0
```

```
Execution address map
```
Address Binding

- How to implement dynamic binding
  - Perform for each address at run time:
    \[ pa = \text{address\_map}(la) \]
  - Simplest form of \text{address\_map}:
    Relocation Register: \[ pa = la + RR \]
  - More general form:
    Page/Segment Table
Third-chance algorithm

- Second chance algorithm does not distinguish between read and write access
- Write access more expensive
- Give modified pages a third chance:
  - \( u \)-bit set at every reference (read and write)
  - \( w \)-bit set at write reference
  - to select a page, cycle through frames, resetting bits, until \( uw == 00 \):

<table>
<thead>
<tr>
<th>( uw \rightarrow uw )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 1 \rightarrow 0 1</td>
</tr>
<tr>
<td>1 0 \rightarrow 0 0</td>
</tr>
</tbody>
</table>
| 0 1 \rightarrow 0 0 *   | (remember modification)
| 0 0 \rightarrow select  |
Third-chance algorithm

- Read→10→00→Select
- Write→11→01→00*→Select

```
... | 0   | 1     | 2     | 3      | 4     | 5     | 6     | 7     | 8     | 9     | 10  .
... | c   | a^w   | d     | b^w    | e     | b     | a^w   | b     | c     | d    .
    ▷ a/10 ▷ a/10 ▷ a/11 ▷ a/11 ▷ a/11 ▷ a/00* ▷ a/00* ▷ a/11 ▷ a/11 ▷ a/00* .
    ▷ b/10 ▷ b/10 ▷ b/10 ▷ b/10 ▷ b/11 ▷ b/00* ▷ b/10* ▷ b/10* ▷ b/10* ▷ d/10 .
    ▷ c/10 ▷ c/10 ▷ c/10 ▷ c/10 ▷ c/10 ▷ e/10 ▷ e/10 ▷ e/10 ▷ e/10 ▷ e/10 ▷ e/00 .
    ▷ d/10 ▷ d/10 ▷ d/10 ▷ d/10 ▷ d/10 ▷ d/00 ▷ d/00 ▷ d/00 ▷ d/00 ▷ c/10 ▷ c/00 .
    ▷ IN  | e     |       |       |       |       |       |       |       |       |       .
    ▷ OUT |       |       |       |       |       |       |       |       |       |       .
```