

SISTEMAS OPERACIONAIS

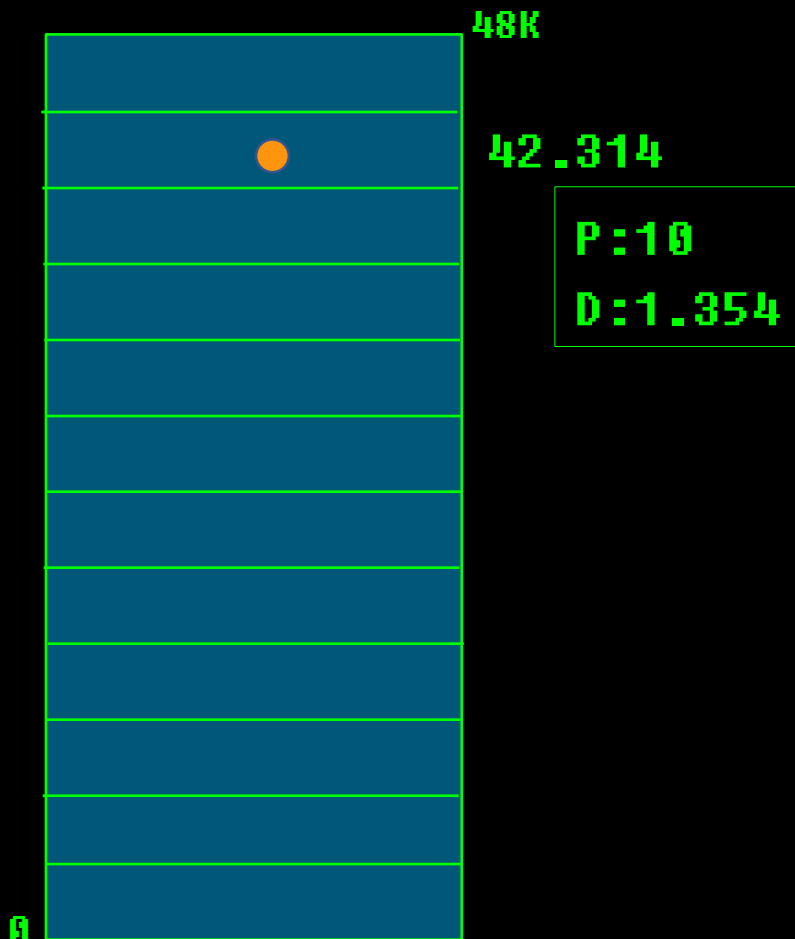
Memória virtual e Substituição de páginas

Prof. Matheus A. Souza
Eng. Software/Sist. Informação
PUC Minas

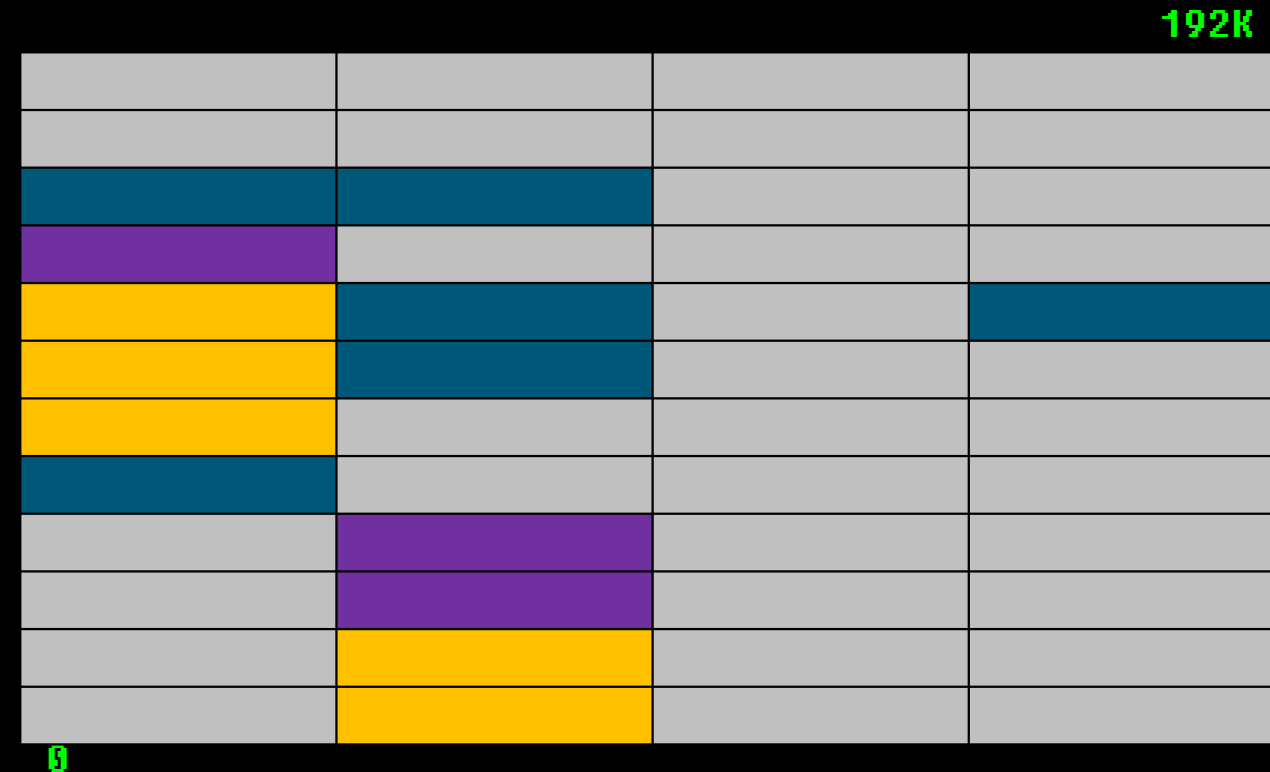
Memória virtual

- > SO mantém em memória principal somente o necessário para a execução dos processos em um dado momento
- > Páginas são carregadas para memória principal de acordo com a demanda do processo
- > Página não presente em memória principal: falta de página (*page fault*)

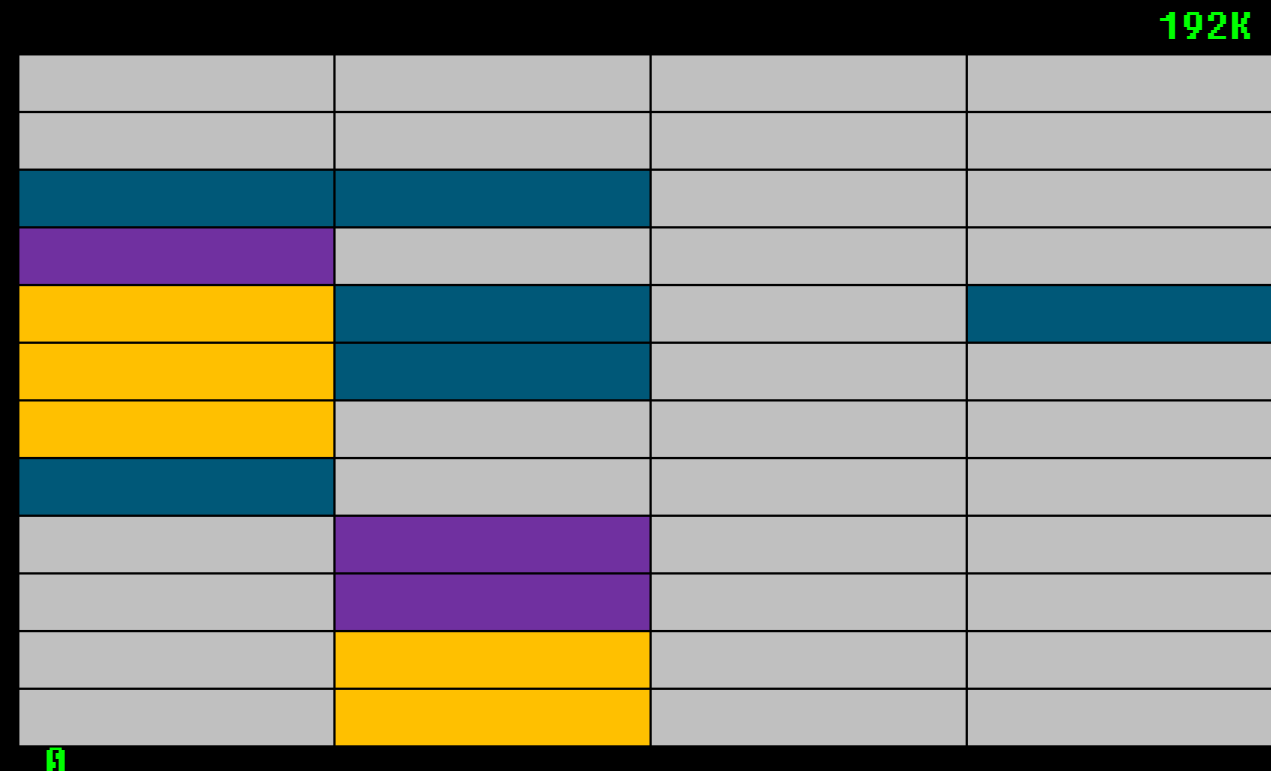
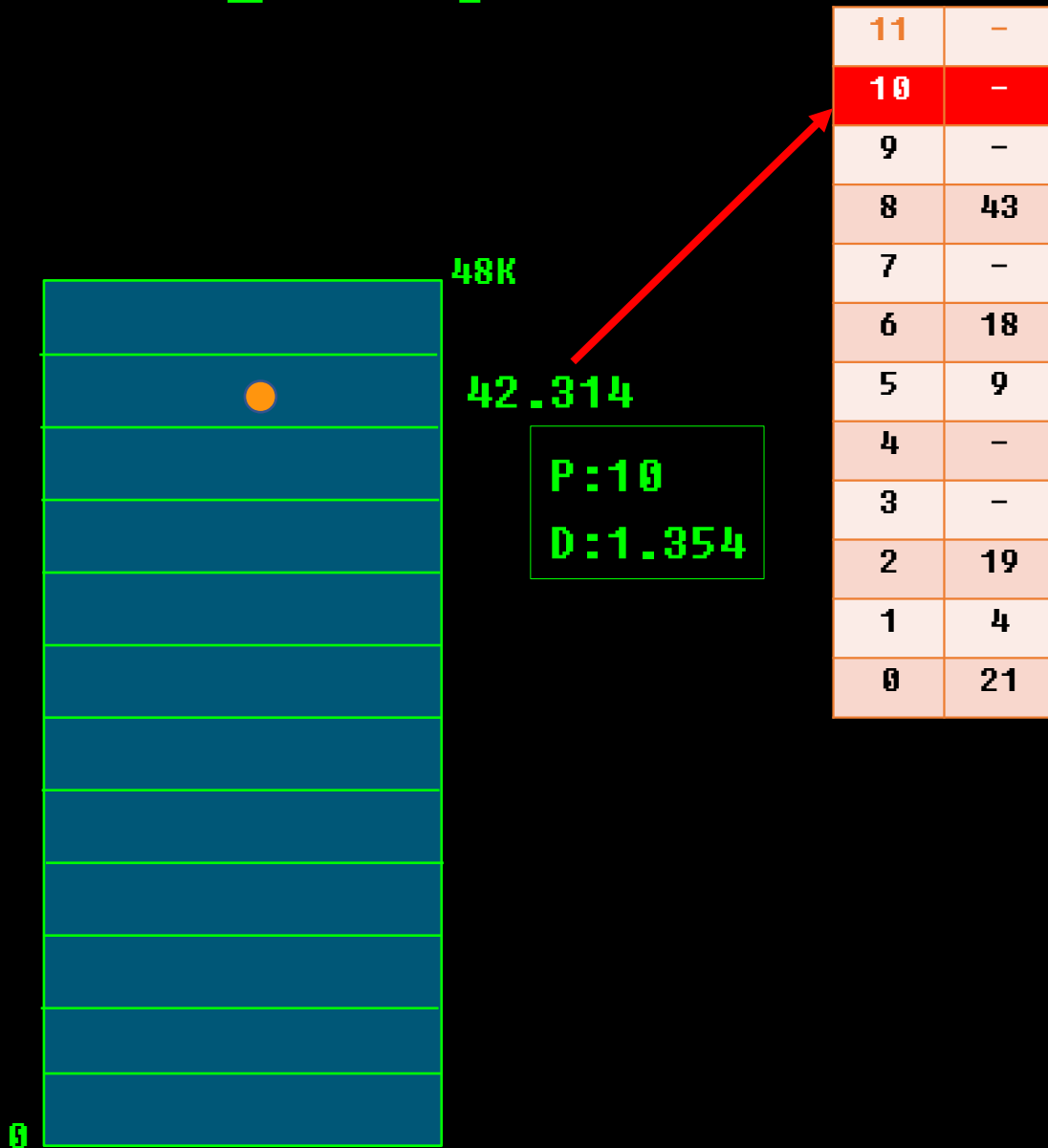
Paginação e memória virtual



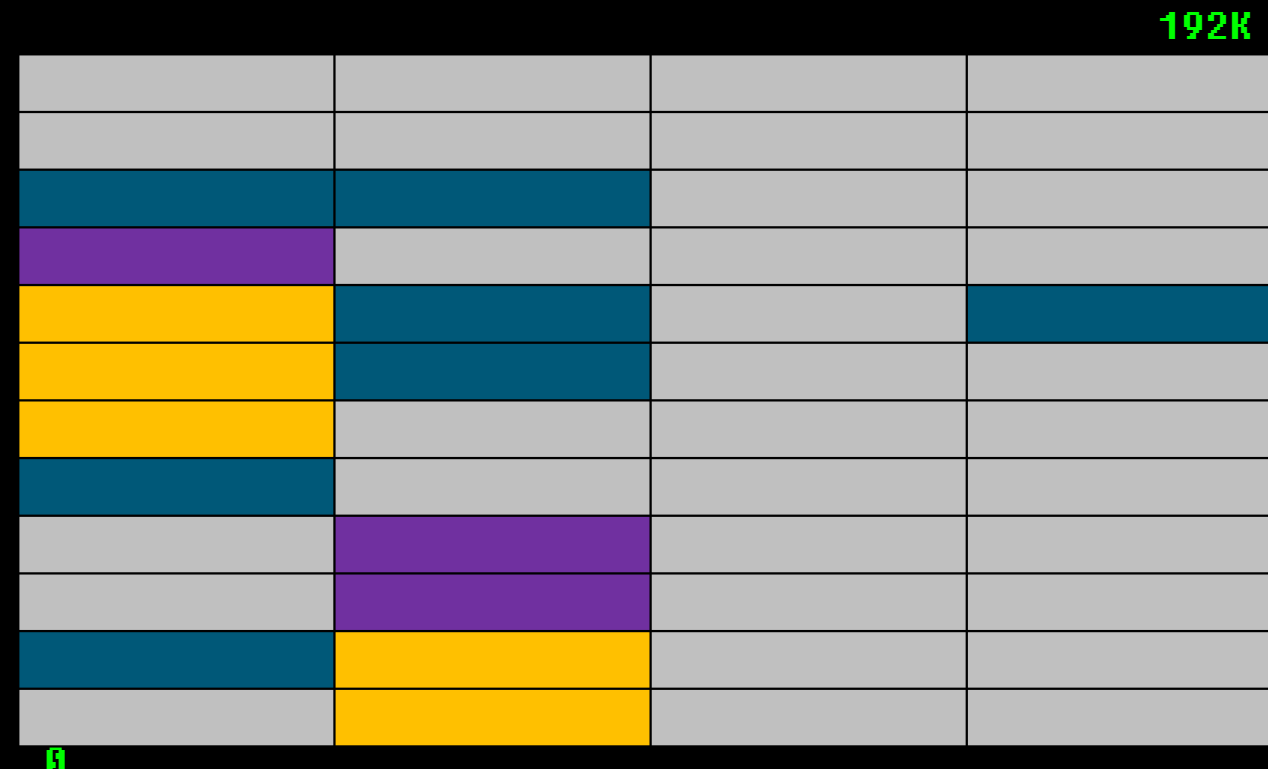
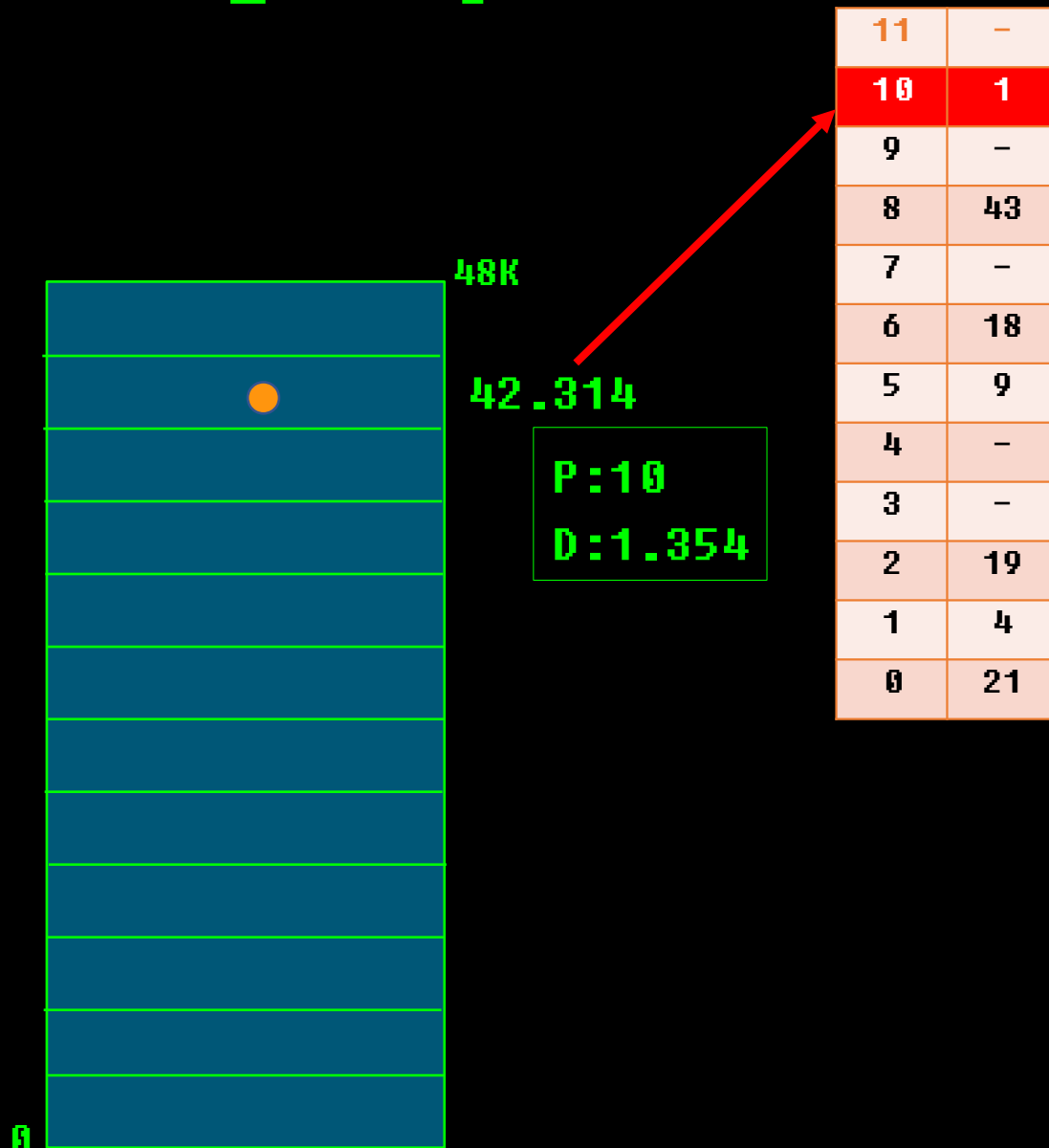
11	-
10	-
9	-
8	43
7	-
6	18
5	9
4	-
3	-
2	19
1	4
0	21



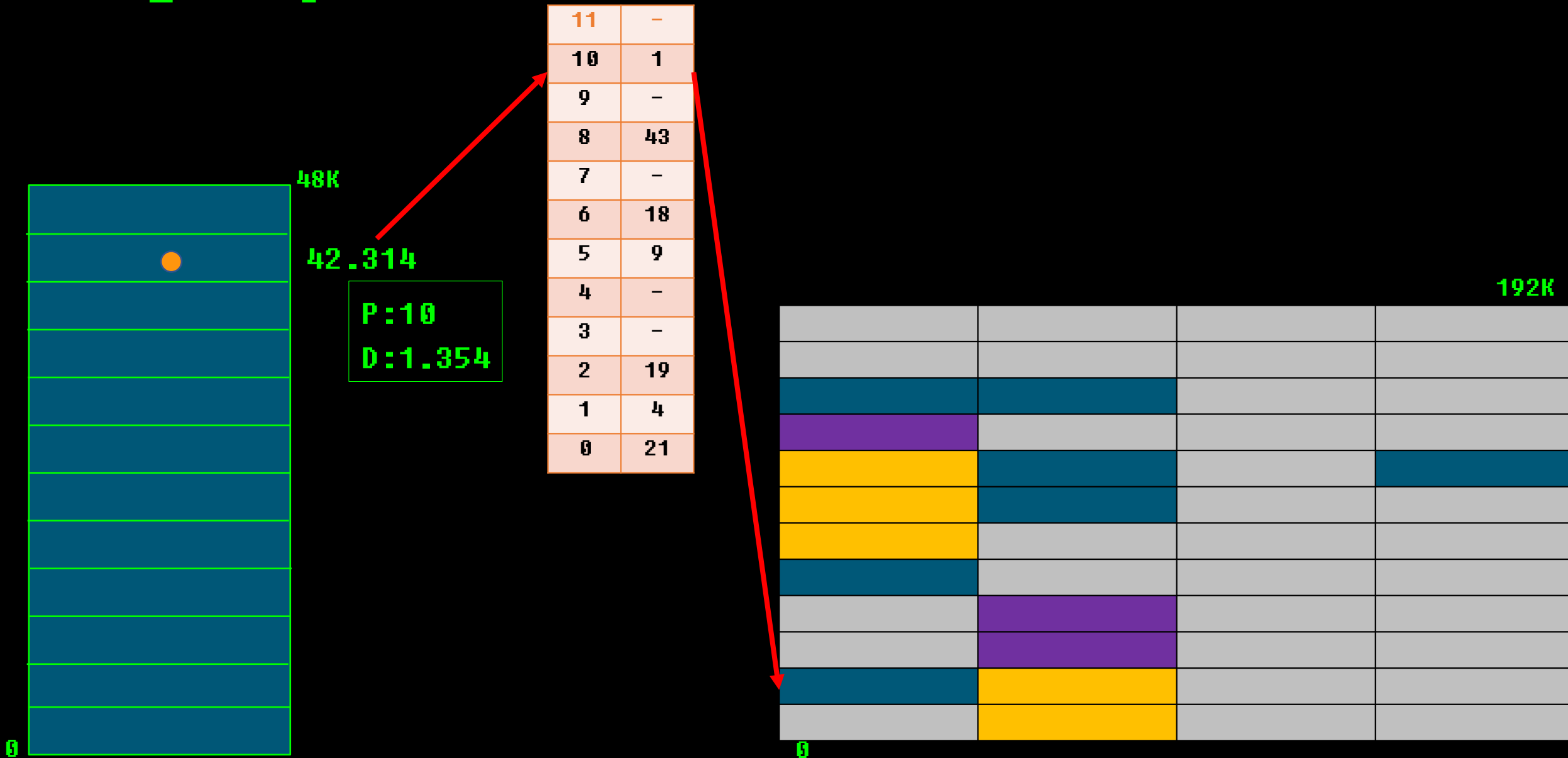
Paginação e memória virtual



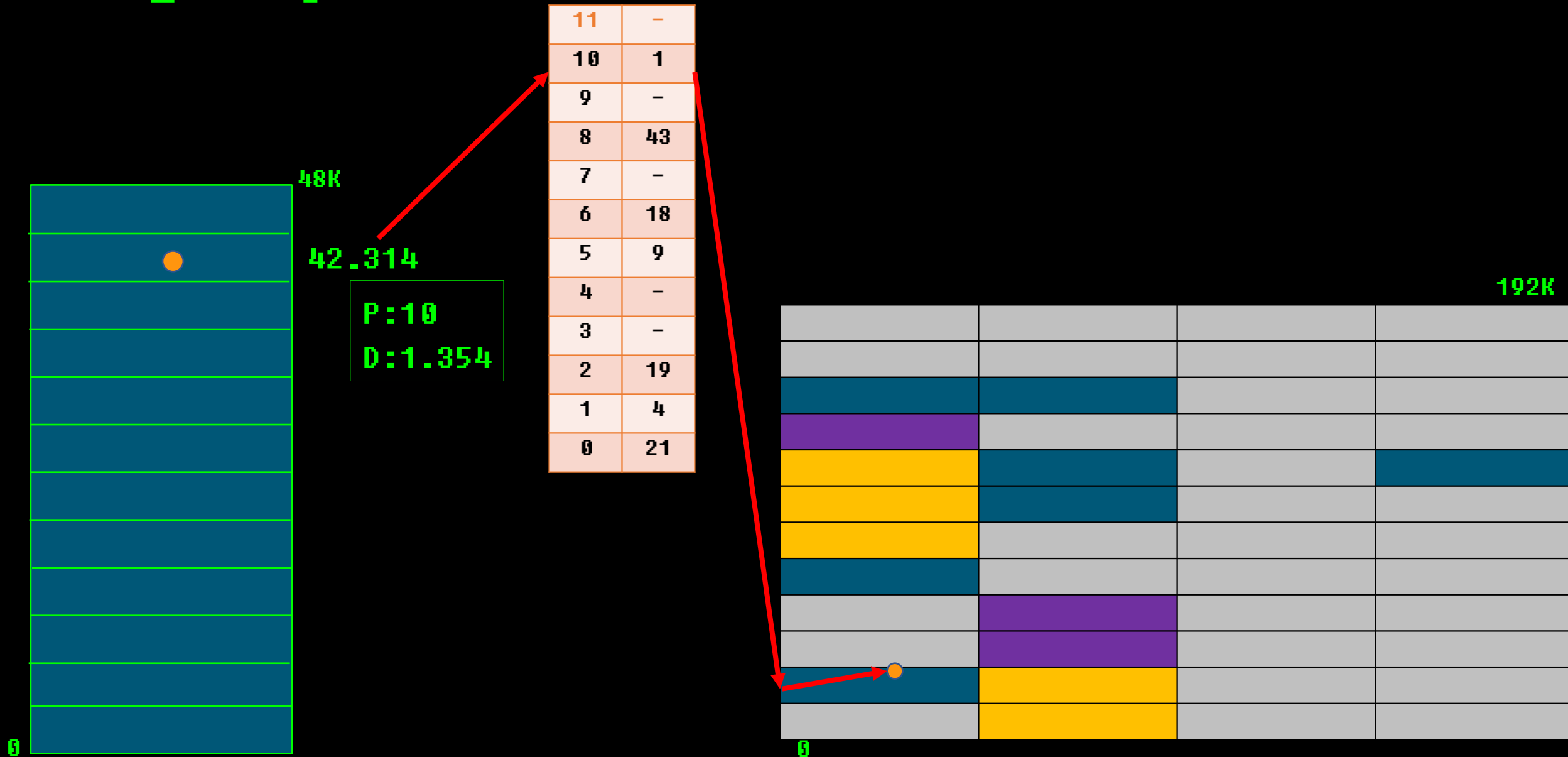
Paginação e memória virtual



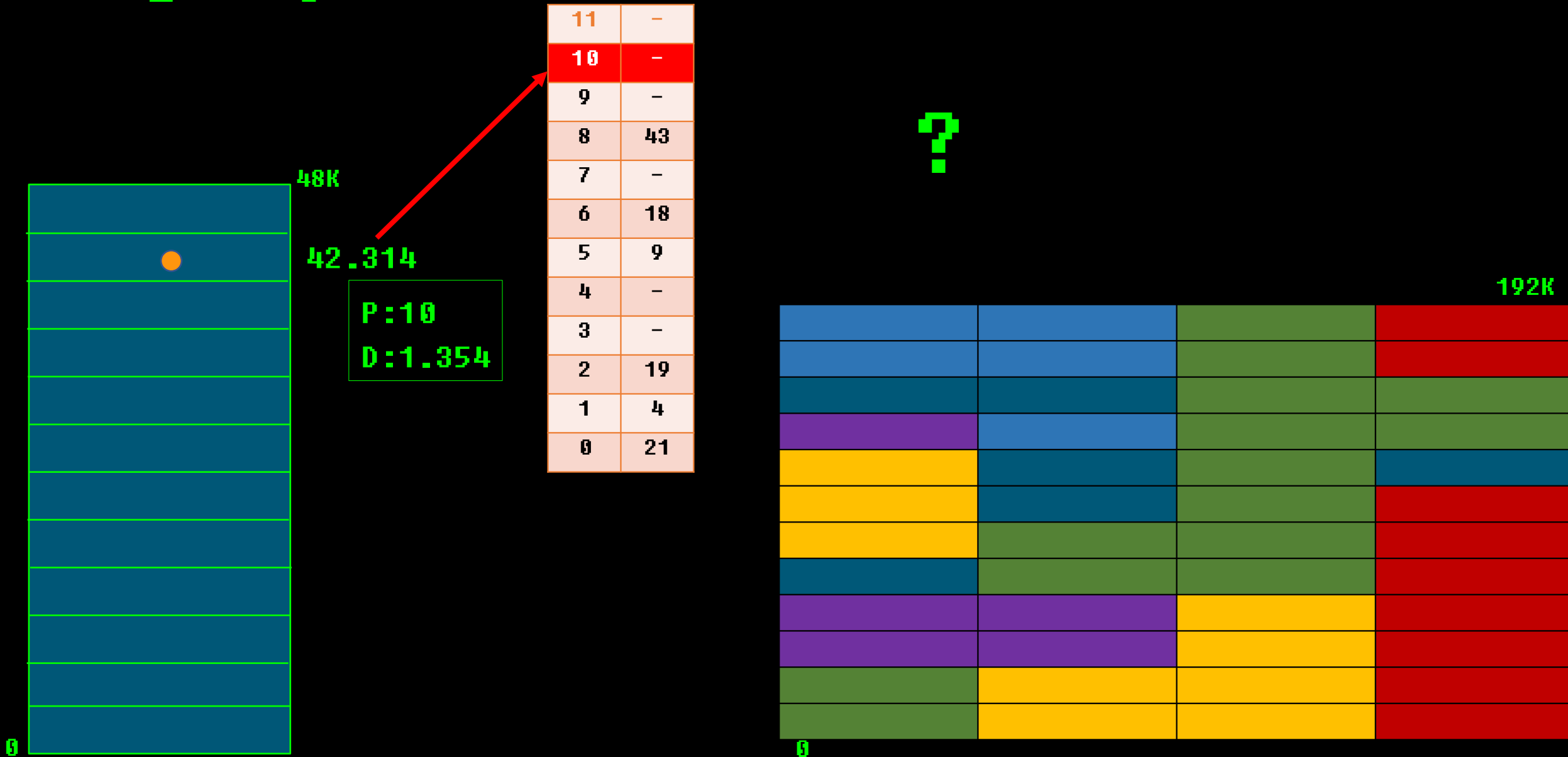
Paginação e memória virtual



Paginação e memória virtual



Paginação e memória virtual



Memória virtual e substituição de páginas

- > Se acontece uma falta de página e não há espaço na memória principal, é necessário *substituir* uma página
- > Algoritmo ótimo: faz a melhor escolha possível para a retirada de uma página.
 - > Qual é esta escolha?

Memória virtual e substituição de páginas

- > Se acontece uma falta de página e não há espaço na memória principal, é necessário *substituir* uma página
- > Algoritmo ótimo: faz a melhor escolha possível para a retirada de uma página.
 - > Qual é esta escolha?



Memória virtual e substituição de páginas

- > Se acontece uma falta de página e não há espaço na memória principal, é necessário *substituir* uma página
- > Algoritmo ótimo: faz a melhor escolha possível para a retirada de uma página.
 - > Qual é esta escolha?
 - > Se não é possível esta escolha, o que fazer?

Algoritmos de substituição de páginas

Algoritmos de substituição

- > Simulação de diferentes regras a partir de uma sequência de acessos de referência
- > Comparação entre si e com o algoritmo ótimo

Algoritmo ótimo

> Memória com 4 molduras.

> Sequência: 3 2 1 0 3 4 3 1 1 5 6 1 0

Algoritmo FIFO

- > A primeira página a entrar será a primeira a sair.
- > Premissa: quem entrou primeiro deixará de ser importante mais cedo.

Algoritmo FIFO

> Memória com 4 molduras.

> Sequência: 3 2 1 0 3 4 3 1 1 5 6 1 0

Algoritmo NRU

> Página Não Recentemente Utilizada
(*Not Recently Used*)

> SO usa dois bits de sinalização

- > R: referência (uso da página)
- > M: modificação (alteração do conteúdo da página)

> Periodicamente, o bit R é zerado

- > Conceito de “não recente”
- > Premissa do algoritmo

> Substituição por classe de prioridade

Classe	R	M
0	0	0
1	0	1
2	1	0
3	1	1

Algoritmo NRU

- > Memória com 4 molduras. Limpeza a cada 4 acessos
 - > Marca de modificação: *
- > Sequência: 3 2* 1 0* 3 4* 3 1 1 5 6 1 0

Algoritmo de Segunda Chance

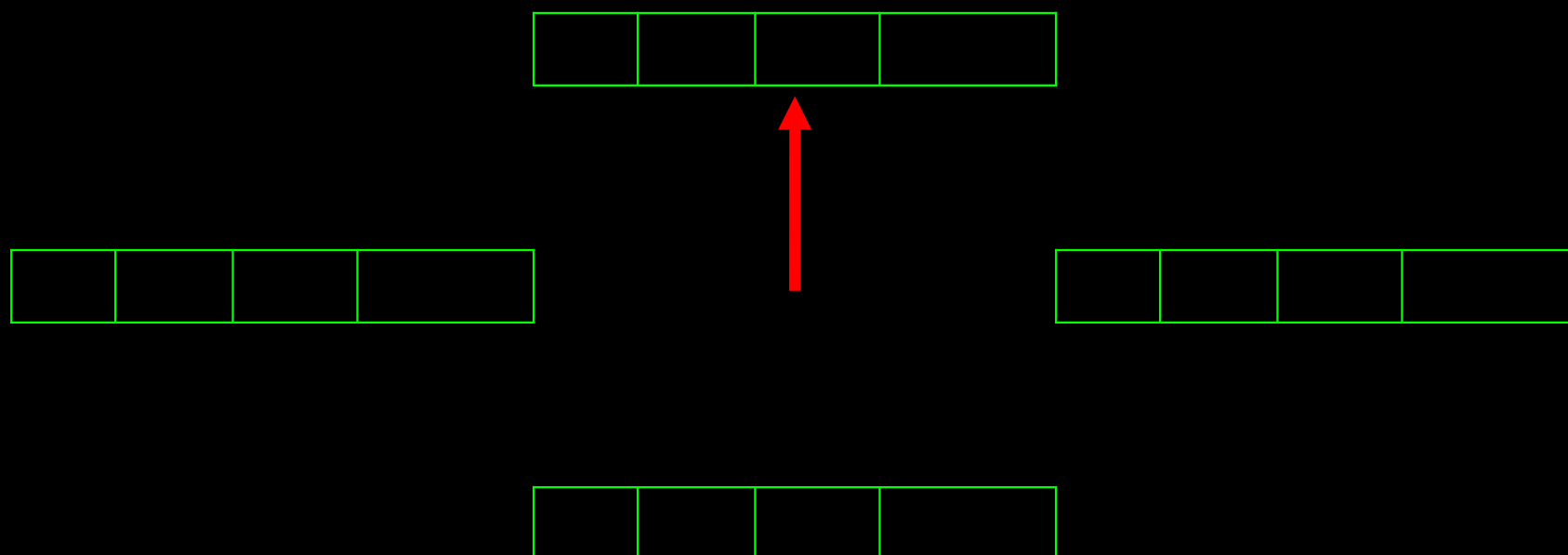
- > Utiliza a lógica de FIFO.
- > Mas a página com bit $R = 1$ ganha uma segunda chance
 - > Zeramos o bit R e colocamos a página no fim da fila
- > Premissa: quem entrou primeiro deixará de ser importante mais cedo, *a não ser que continue sendo utilizada.*

Algoritmo de Segunda Chance / Relógio

- > Segunda chance é geralmente implementada com uma lista circular, por questão de desempenho
- > Algoritmo do *relógio*

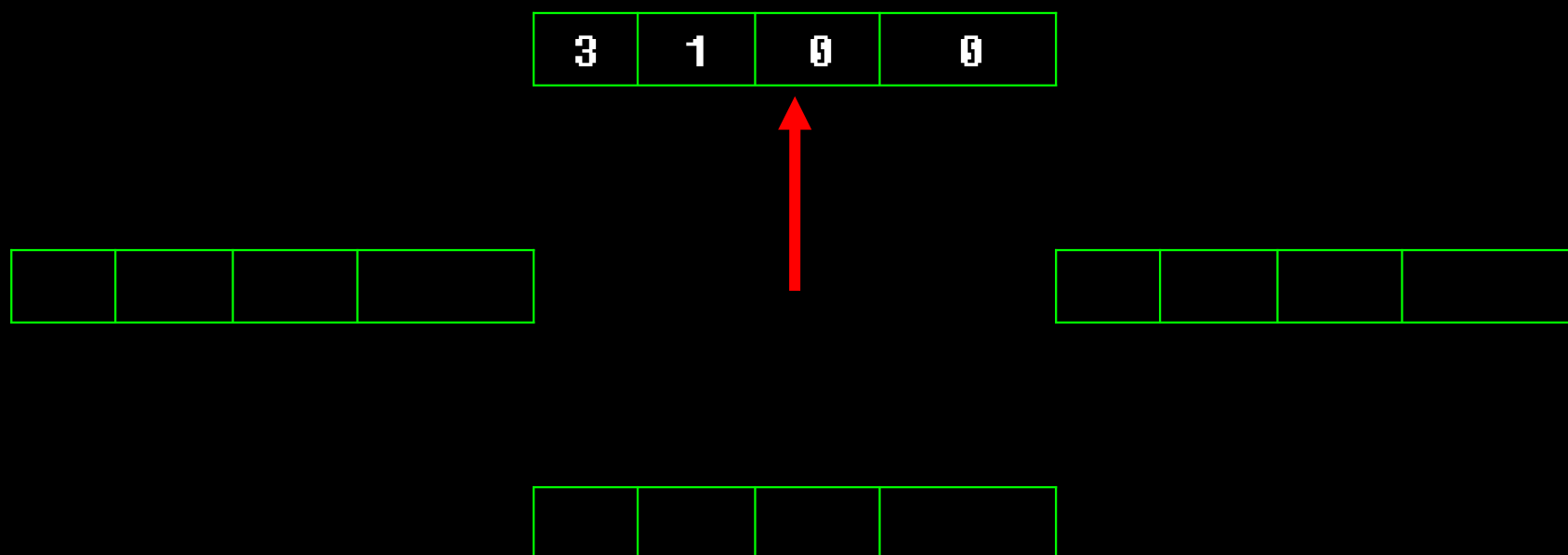
Algoritmo de Segunda Chance / Relógio

- > Algoritmo do *relógio*
- > Memória com 4 molduras. Limpeza a cada 4 acessos
 - > Marca de modificação: *
- > Sequência: 3 2* 1 0* 3 4* 3 1 1 5 6 1 0



Algoritmo de Segunda Chance / Relógio

- > Algoritmo do *relógio*
- > Memória com 4 molduras. Limpeza a cada 4 acessos
 - > Marca de modificação: *
- > Sequência: 3 2* 1 0* 3 4* 3 1 1 5 6 1 0



Algoritmo de Segunda Chance / Relógio

- > Algoritmo do *relógio*
- > Memória com 4 molduras. Limpeza a cada 4 acessos
 - > Marca de modificação: *
- > Sequência: 3 2* 1 0* 3 4* 3 1 1 5 6 1 0

3	1	0	0
---	---	---	---

--	--	--	--

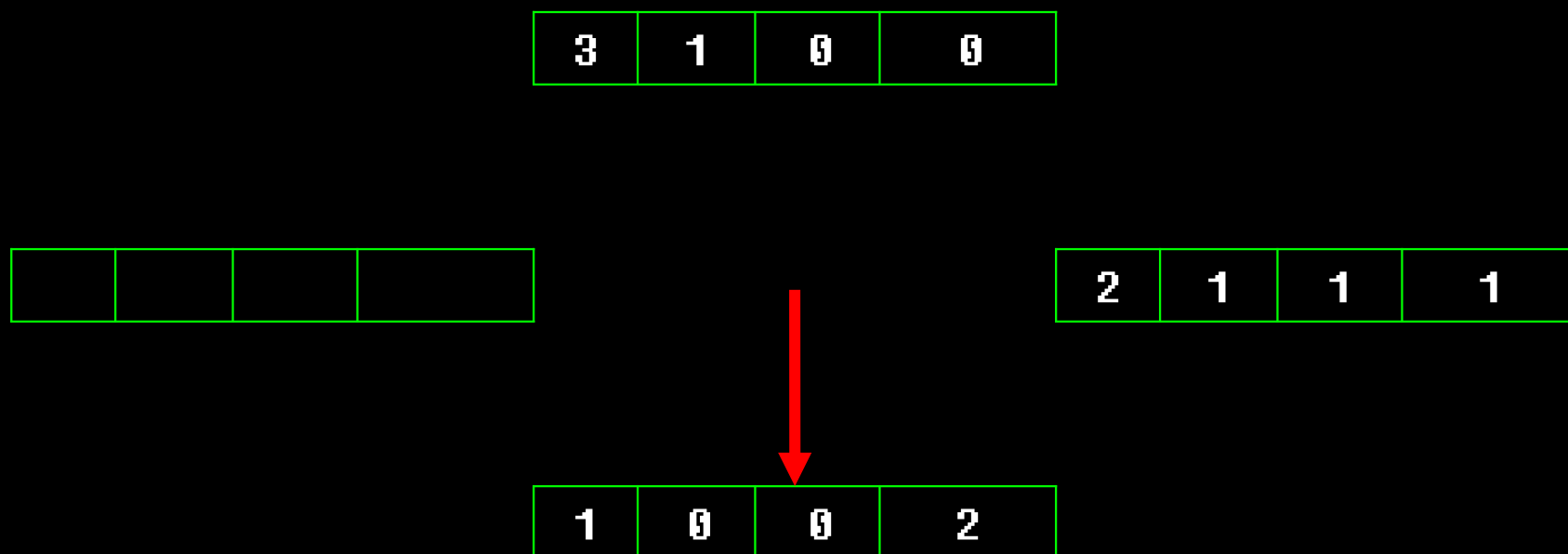


2	1	1	1
---	---	---	---

--	--	--	--

Algoritmo de Segunda Chance / Relógio

- > Algoritmo do *relógio*
- > Memória com 4 molduras. Limpeza a cada 4 acessos
 - > Marca de modificação: *
- > Sequência: 3 2* 1 0* 3 4* 3 1 1 5 6 1 0



Algoritmo de Segunda Chance / Relógio

- > Algoritmo do *relógio*
- > Memória com 4 molduras. Limpeza a cada 4 acessos
 - > Marca de modificação: *
- > Sequência: 3 2* 1 0* 3 4* 3 1 1 5 6 1 0

3	1	0	0
---	---	---	---

0	1	1	3
---	---	---	---

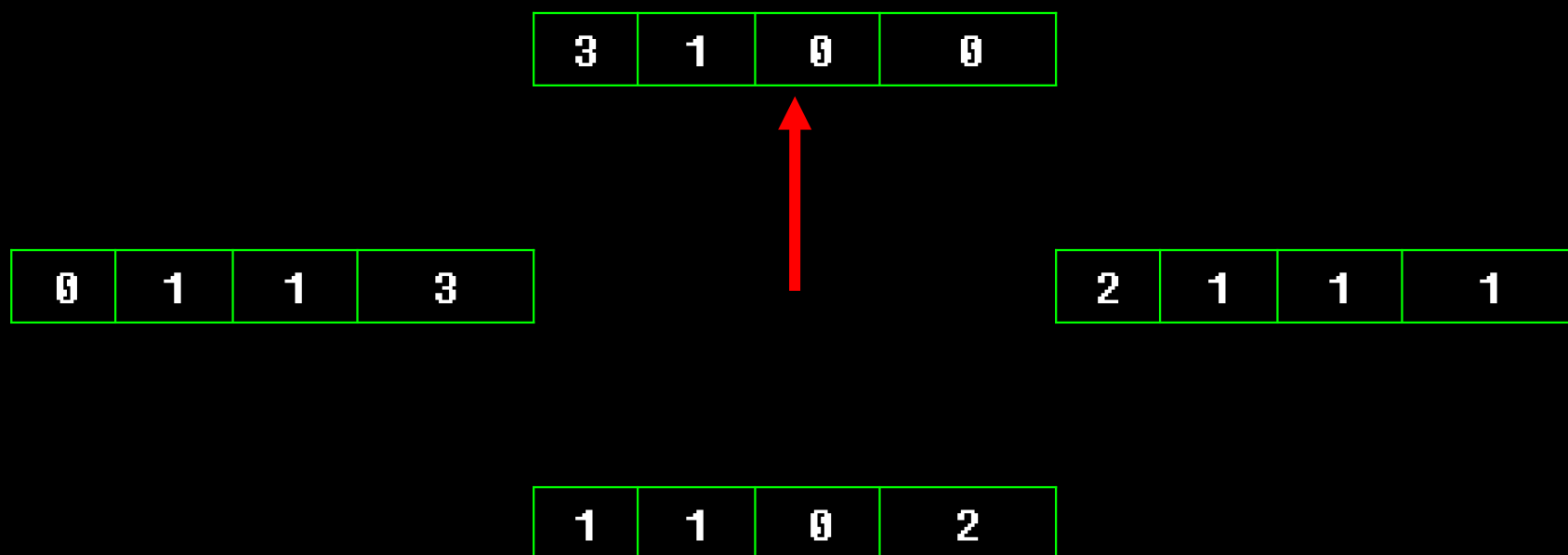


2	1	1	1
---	---	---	---

1	1	0	2
---	---	---	---

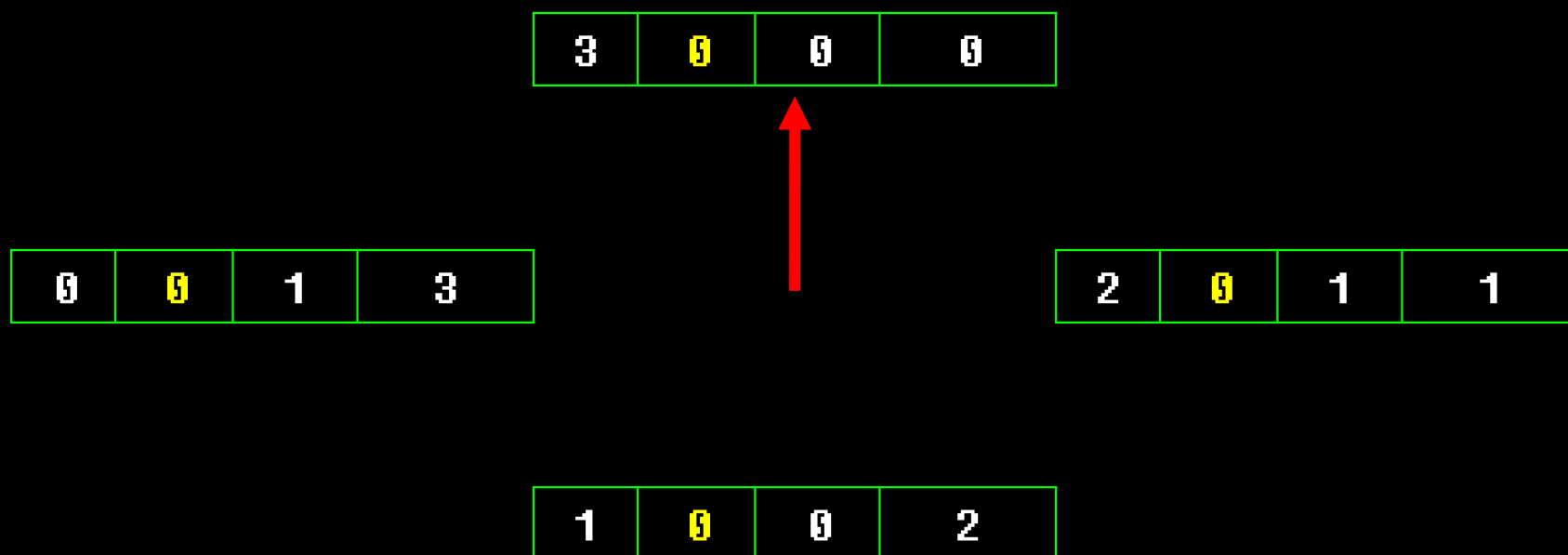
Algoritmo de Segunda Chance / Relógio

- > Algoritmo do *relógio*
- > Memória com 4 molduras. Limpeza a cada 4 acessos
 - > Marca de modificação: *
- > Sequência: 3 2* 1 0* 3 4* 3 1 1 5 6 1 0



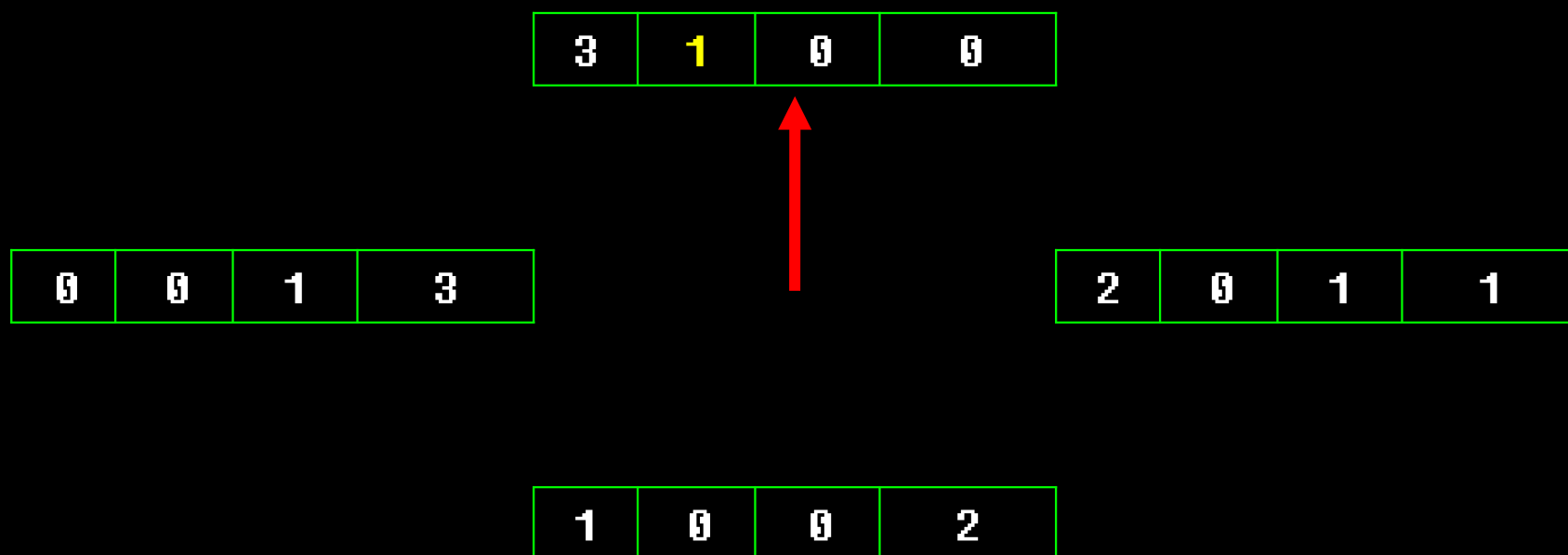
Algoritmo de Segunda Chance / Relógio

- > Algoritmo do *relógio*
- > Memória com 4 molduras. Limpeza a cada 4 acessos
 - > Marca de modificação: *
- > Sequência: ~~3~~ ~~2*~~ ~~1~~ ~~0*~~ 3 4* 3 1 1 5 6 1 0



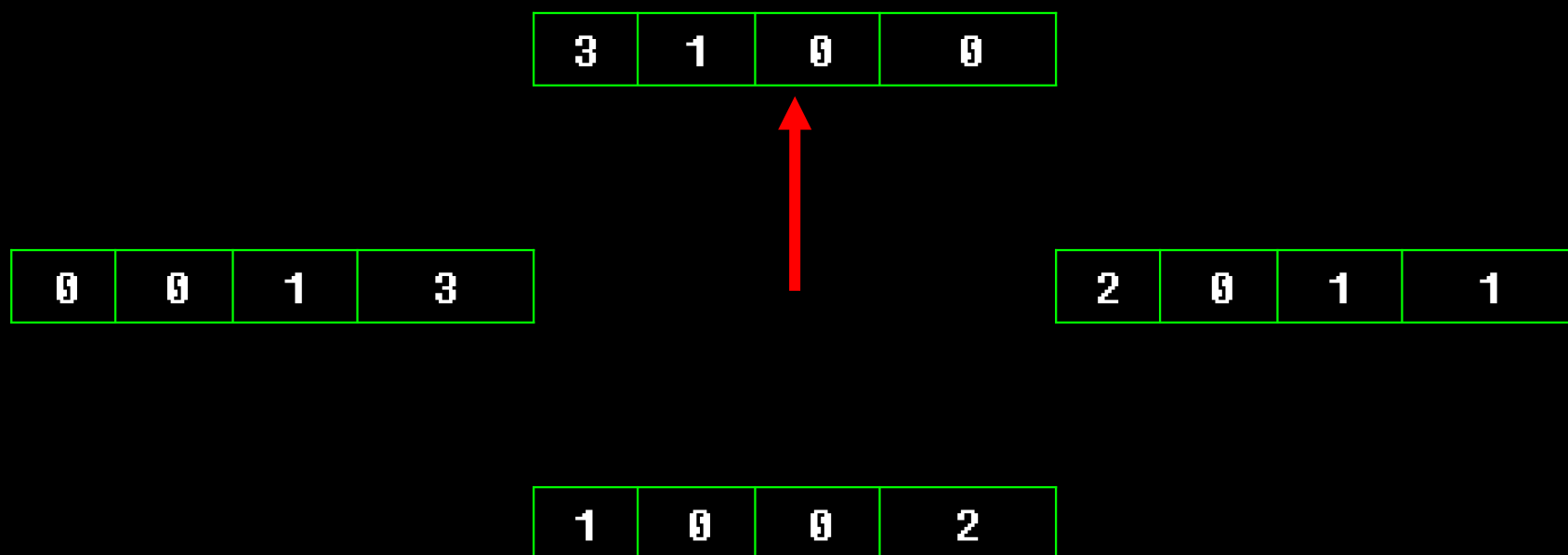
Algoritmo de Segunda Chance / Relógio

- > Algoritmo do *relógio*
- > Memória com 4 molduras. Limpeza a cada 4 acessos
 - > Marca de modificação: *
- > Sequência: ~~3~~ ~~2*~~ ~~1~~ ~~0*~~ 3 4* 3 1 1 5 6 1 0



Algoritmo de Segunda Chance / Relógio

- > Algoritmo do *relógio*
- > Memória com 4 molduras. Limpeza a cada 4 acessos
 - > Marca de modificação: *
- > Sequência: ~~3~~ ~~2*~~ ~~1~~ ~~0*~~ 3 4* 3 1 1 5 6 1 0



Algoritmo de Segunda Chance / Relógio

- > Algoritmo do *relógio*
- > Memória com 4 molduras. Limpeza a cada 4 acessos
 - > Marca de modificação: *
- > Sequência: ~~3~~ ~~2*~~ ~~1~~ ~~0*~~ 3 4* 3 1 1 5 6 1 0

3	0	0	0
---	---	---	---

0	0	1	3
---	---	---	---



2	0	1	1
---	---	---	---

1	0	0	2
---	---	---	---

Algoritmo de Segunda Chance / Relógio

- > Algoritmo do *relógio*
- > Memória com 4 molduras. Limpeza a cada 4 acessos
 - > Marca de modificação: *
- > Sequência: ~~0~~ ~~2*~~ ~~1~~ ~~0*~~ 3 4* 3 1 1 5 6 1 0

3	0	0	0
---	---	---	---

0	0	1	3
---	---	---	---



4	1	1	1
---	---	---	---

1	0	0	2
---	---	---	---

Algoritmo de Segunda Chance / Relógio

- > Algoritmo do *relógio*
- > Memória com 4 molduras. Limpeza a cada 4 acessos
 - > Marca de modificação: *
- > Sequência: ~~3~~ ~~2*~~ ~~1~~ ~~0*~~ 3 4* 3 1 1 5 6 1 0

3	0	0	0
---	---	---	---

0	0	1	3
---	---	---	---

4	1	1	1
---	---	---	---



1	0	0	2
---	---	---	---

Algoritmo de Segunda Chance / Relógio

- > Algoritmo do *relógio*
- > Memória com 4 molduras. Limpeza a cada 4 acessos
 - > Marca de modificação: *
- > Sequência: ~~3~~ ~~2*~~ ~~1~~ ~~0*~~ 3 4* 3 1 1 5 6 1 0

3	1	0	0
---	---	---	---

0	0	1	3
---	---	---	---

4	1	1	1
---	---	---	---



1	0	0	2
---	---	---	---

Algoritmo de Segunda Chance / Relógio

- > Algoritmo do *relógio*
- > Memória com 4 molduras. Limpeza a cada 4 acessos
 - > Marca de modificação: *
- > Sequência: ~~3~~ ~~2*~~ ~~1~~ ~~0*~~ 3 4* 3 1 1 5 6 1 0

3	1	0	0
---	---	---	---

0	0	1	3
---	---	---	---

4	1	1	1
---	---	---	---

1	1	0	2
---	---	---	---



Algoritmo de Segunda Chance / Relógio

- > Algoritmo do *relógio*
- > Memória com 4 molduras. Limpeza a cada 4 acessos
 - > Marca de modificação: *
- > Sequência: ~~3~~ ~~2*~~ ~~1~~ ~~0*~~ ~~3~~ ~~4*~~ ~~3~~ ~~1~~ 1 5 6 1 0

3	0	0	0
---	---	---	---

0	0	1	3
---	---	---	---

4	0	1	1
---	---	---	---



1	0	0	2
---	---	---	---

Algoritmo de Segunda Chance / Relógio

- > Algoritmo do *relógio*
- > Memória com 4 molduras. Limpeza a cada 4 acessos
 - > Marca de modificação: *
- > Sequência: ~~3~~ ~~2*~~ ~~1~~ ~~0*~~ ~~3~~ ~~4*~~ ~~3~~ ~~1~~ 1 5 6 1 0

3	0	0	0
---	---	---	---

0	0	1	3
---	---	---	---

4	0	1	1
---	---	---	---

1	1	0	2
---	---	---	---

Algoritmo de Segunda Chance / Relógio

- > Algoritmo do *relógio*
- > Memória com 4 molduras. Limpeza a cada 4 acessos
 - > Marca de modificação: *
- > Sequência: ~~3~~ ~~2*~~ ~~1~~ ~~0*~~ ~~3~~ ~~4*~~ ~~3~~ ~~1~~ 1 5 6 1 0

3	0	0	0
---	---	---	---

0	0	1	3
---	---	---	---

4	0	1	1
---	---	---	---

1	1	0	2
---	---	---	---



Algoritmo de Segunda Chance / Relógio

- > Algoritmo do *relógio*
- > Memória com 4 molduras. Limpeza a cada 4 acessos
 - > Marca de modificação: *
- > Sequência: ~~3~~ ~~2*~~ ~~1~~ ~~0*~~ ~~3~~ ~~4*~~ ~~3~~ ~~1~~ 1 5 6 1 0

3	0	0	0
---	---	---	---

0	0	1	3
---	---	---	---



4	0	1	1
---	---	---	---

1	0	0	2
---	---	---	---

Algoritmo de Segunda Chance / Relógio

- > Algoritmo do *relógio*
- > Memória com 4 molduras. Limpeza a cada 4 acessos
 - > Marca de modificação: *
- > Sequência: ~~3~~ ~~2*~~ ~~1~~ ~~0*~~ ~~3~~ ~~4*~~ ~~3~~ ~~1~~ 1 5 6 1 0

3	0	0	0
---	---	---	---

5	1	0	3
---	---	---	---

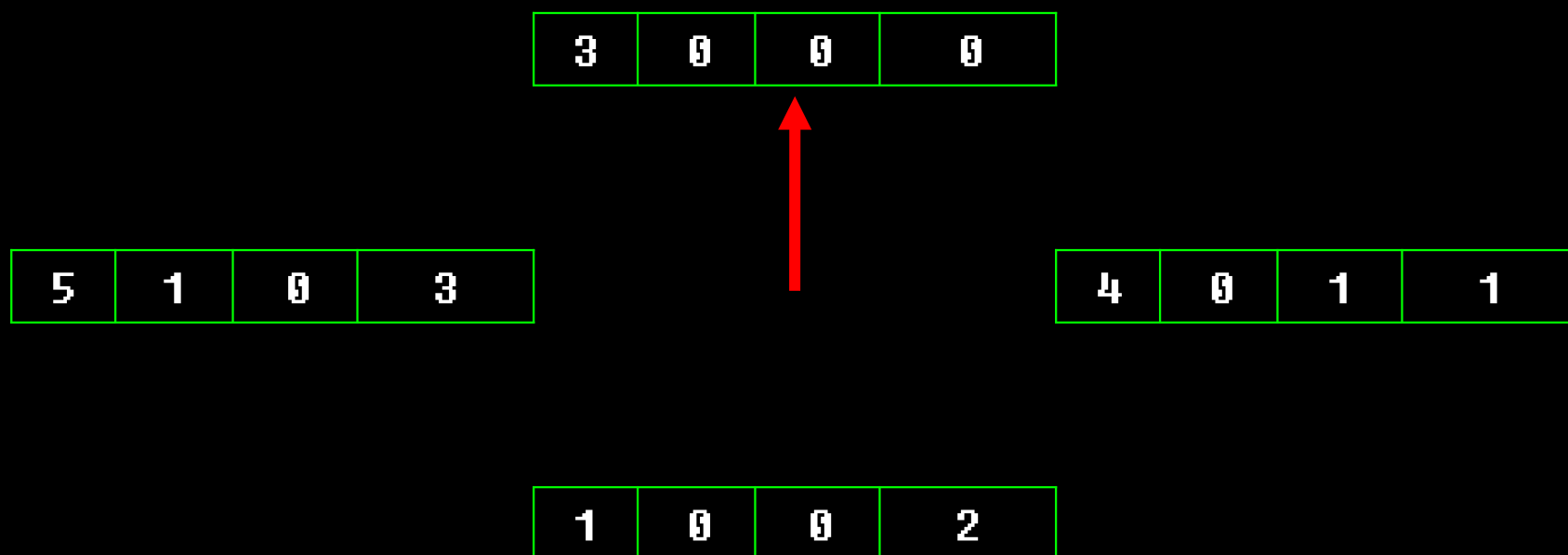


4	0	1	1
---	---	---	---

1	0	0	2
---	---	---	---

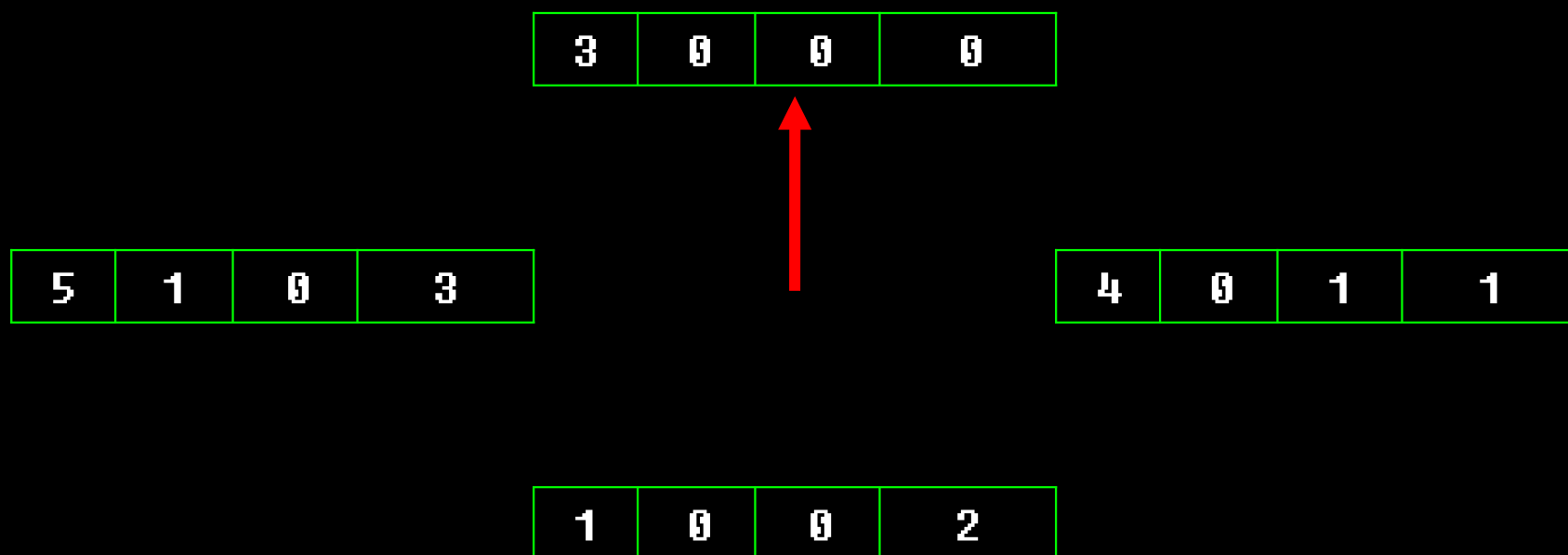
Algoritmo de Segunda Chance / Relógio

- > Algoritmo do *relógio*
- > Memória com 4 molduras. Limpeza a cada 4 acessos
 - > Marca de modificação: *
- > Sequência: ~~3~~ ~~2*~~ ~~1~~ ~~0*~~ ~~3~~ ~~4*~~ ~~3~~ ~~1~~ 1 5 6 1 0



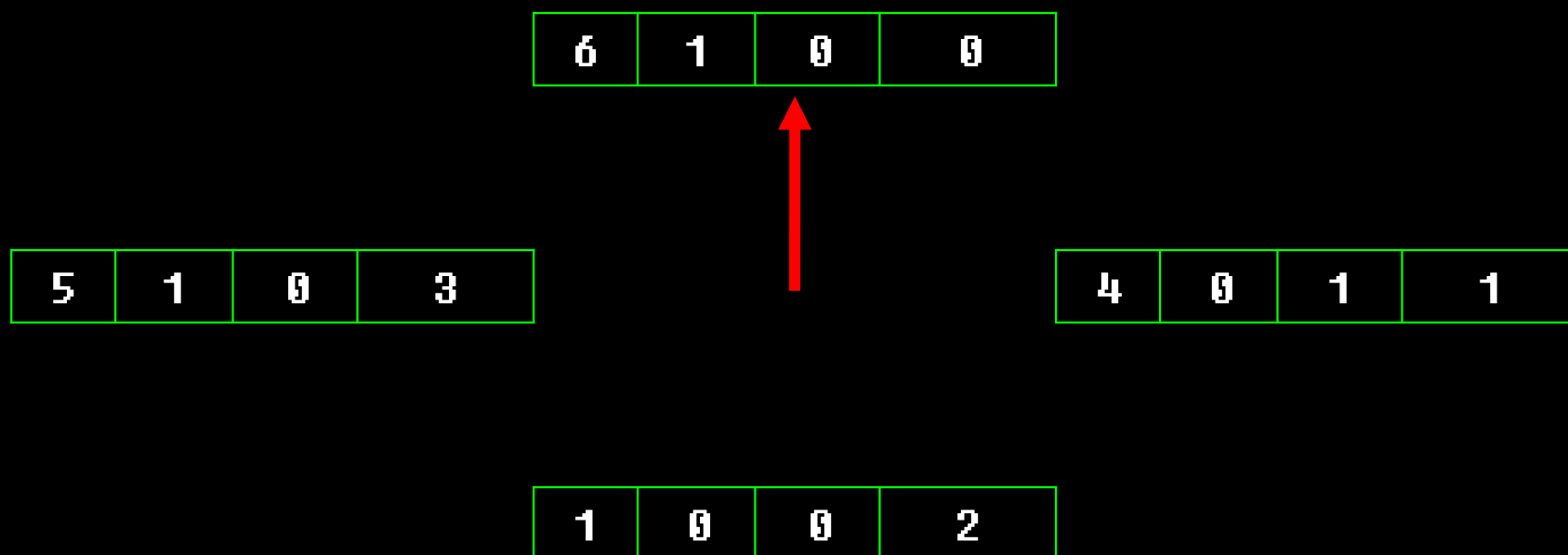
Algoritmo de Segunda Chance / Relógio

- > Algoritmo do *relógio*
- > Memória com 4 molduras. Limpeza a cada 4 acessos
 - > Marca de modificação: *
- > Sequência: ~~3~~ ~~2*~~ ~~1~~ ~~0*~~ ~~3~~ ~~4*~~ ~~3~~ ~~1~~ 1 5 **6** 1 0



Algoritmo de Segunda Chance / Relógio

- > Algoritmo do *relógio*
- > Memória com 4 molduras. Limpeza a cada 4 acessos
 - > Marca de modificação: *
- > Sequência: ~~3~~ ~~2*~~ ~~1~~ ~~0*~~ ~~3~~ ~~4*~~ ~~3~~ ~~1~~ 1 5 6 1 0



Algoritmo de Segunda Chance / Relógio

- > Algoritmo do *relógio*
- > Memória com 4 molduras. Limpeza a cada 4 acessos
 - > Marca de modificação: *
- > Sequência: ~~3~~ ~~2*~~ ~~1~~ ~~0*~~ ~~3~~ ~~4*~~ ~~3~~ ~~1~~ 1 5 6 1 0

6	1	0	0
---	---	---	---

5	1	0	3
---	---	---	---



4	0	1	1
---	---	---	---

1	0	0	2
---	---	---	---

Algoritmo de Segunda Chance / Relógio

- > Algoritmo do *relógio*
- > Memória com 4 molduras. Limpeza a cada 4 acessos
 - > Marca de modificação: *
- > Sequência: ~~3~~ ~~2*~~ ~~1~~ ~~0*~~ ~~3~~ ~~4*~~ ~~3~~ ~~1~~ 1 5 6 1 0

6	1	0	0
---	---	---	---

5	1	0	3
---	---	---	---



4	0	1	1
---	---	---	---

1	1	0	2
---	---	---	---

Algoritmo de Segunda Chance / Relógio

- > Algoritmo do *relógio*
- > Memória com 4 molduras. Limpeza a cada 4 acessos
 - > Marca de modificação: *
- > Sequência: ~~3~~ ~~2*~~ ~~1~~ ~~0*~~ ~~3~~ ~~4*~~ ~~3~~ ~~1~~ ~~1~~ ~~5~~ ~~6~~ ~~1~~ ~~0~~

6	0	0	0
---	---	---	---

5	0	0	3
---	---	---	---



4	0	1	1
---	---	---	---

1	0	0	2
---	---	---	---

Algoritmo de Segunda Chance / Relógio

- > Algoritmo do *relógio*
- > Memória com 4 molduras. Limpeza a cada 4 acessos
 - > Marca de modificação: *
- > Sequência: ~~3~~ ~~2*~~ ~~1~~ ~~0*~~ ~~3~~ ~~4*~~ ~~3~~ ~~1~~ ~~1~~ ~~5~~ ~~6~~ ~~1~~ **0**

6	0	0	0
---	---	---	---

5	0	0	3
---	---	---	---



4	0	1	1
---	---	---	---

1	0	0	2
---	---	---	---

Algoritmo de Segunda Chance / Relógio

- > Algoritmo do *relógio*
- > Memória com 4 molduras. Limpeza a cada 4 acessos
 - > Marca de modificação: *
- > Sequência: ~~3~~ ~~2*~~ ~~1~~ ~~0*~~ ~~3~~ ~~4*~~ ~~3~~ ~~1~~ ~~1~~ ~~5~~ ~~6~~ ~~1~~ **0**

6	0	0	0
---	---	---	---

5	0	0	3
---	---	---	---



0	1	0	1
---	---	---	---

1	0	0	2
---	---	---	---

Algoritmo de Segunda Chance / Relógio

- > Algoritmo do *relógio*
- > Memória com 4 molduras. Limpeza a cada 4 acessos
 - > Marca de modificação: *
- > Sequência: ~~3~~ ~~2*~~ ~~1~~ ~~0*~~ ~~3~~ ~~4*~~ ~~3~~ ~~1~~ ~~1~~ ~~5~~ ~~6~~ ~~1~~ ~~0~~

6	0	0	0
---	---	---	---

5	0	0	3
---	---	---	---

0	1	0	1
---	---	---	---

1	0	0	2
---	---	---	---



Algoritmo LFU

- > Página menos frequentemente utilizada (*Least Frequently Used*)
- > Contador de acessos a cada página: sai a que tiver menor contador
- > Premissa: uma página pouco usada até o momento provavelmente será pouco importante no futuro.

Algoritmo LFU

- > Página menos frequentemente utilizada (*Least Frequently Used*)
- > Contador de acessos a cada página: sai a que tiver menor contador
- > É barato manter o contador?
- > E se a página acabou de entrar, qual será seu contador?

Algoritmo LFU com envelhecimento

- > Simulação do contador real: conta *intervalos* de uso
- > Envelhecimento: tentativa de “esquecer” valores de acessos antigos
 - > A página é importante se continuar sendo utilizada
- > Uso do bit R para implementação

Algoritmo LFU com envelhecimento

- > Uso do bit R para implementação
- > Conjunto de bits de referência: “idade”
 - > Bit R é o bit à esquerda da “idade”
- > A cada época, *shift* à direita da idade
 - > *Envelhecimento*
- > Sai a página com menor valor de “idade” (menos uso, ou usos mais antigos)

Algoritmo LFU com envelhecimento

- > Memória com 4 molduras. Idade de 4 bits.
- > Limpeza a cada 4 acessos
- > Sequência: 3 2 1 0 3 4 3 1 1 5 6 1 0

Página	Idade

Algoritmo LFU com envelhecimento

- > Memória com 4 molduras. Idade de 4 bits.
- > Limpeza a cada 4 acessos
- > Sequência: 3 2 1 0 3 4 3 1 1 5 6 1 0

Página	Idade
3	1000

Algoritmo LFU com envelhecimento

- > Memória com 4 molduras. Idade de 4 bits.
- > Limpeza a cada 4 acessos
- > Sequência: 3 2 1 0 3 4 3 1 1 5 6 1 0

Página	Idade
3	1000
2	1000

Algoritmo LFU com envelhecimento

- > Memória com 4 molduras. Idade de 4 bits.
- > Limpeza a cada 4 acessos
- > Sequência: 3 2 1 0 3 4 3 1 1 5 6 1 0

Página	Idade
3	1000
2	1000
1	1000

Algoritmo LFU com envelhecimento

- > Memória com 4 molduras. Idade de 4 bits.
- > Limpeza a cada 4 acessos
- > Sequência: 3 2 1 0 3 4 3 1 1 5 6 1 0

Página	Idade
3	1000
2	1000
1	1000
0	1000

Algoritmo LFU com envelhecimento

- > Memória com 4 molduras. Idade de 4 bits.
- > Limpeza a cada 4 acessos
- > Sequência: ~~3~~ ~~2~~ ~~1~~ ~~0~~ 3 4 3 1 1 5 6 1 0

Página	Idade
3	0100
2	0100
1	0100
0	0100

Algoritmo LFU com envelhecimento

- > Memória com 4 molduras. Idade de 4 bits.
- > Limpeza a cada 4 acessos
- > Sequência: ~~3~~ ~~2~~ ~~1~~ ~~0~~ 3 4 3 1 1 5 6 1 0

Página	Idade
3	0 100
2	0100
1	0100
0	0100

Algoritmo LFU com envelhecimento

- > Memória com 4 molduras. Idade de 4 bits.
- > Limpeza a cada 4 acessos
- > Sequência: ~~3~~ ~~2~~ ~~1~~ ~~0~~ 3 4 3 1 1 5 6 1 0

Página	Idade
3	1100
2	0100
1	0100
0	0100

Algoritmo LFU com envelhecimento

- > Memória com 4 molduras. Idade de 4 bits.
- > Limpeza a cada 4 acessos
- > Sequência: ~~3~~ ~~2~~ ~~1~~ ~~0~~ 3 4 3 1 1 5 6 1 0

Página	Idade
3	1100
4	1000
1	0100
0	0100

Algoritmo LFU com envelhecimento

- > Memória com 4 molduras. Idade de 4 bits.
- > Limpeza a cada 4 acessos
- > Sequência: ~~3~~ ~~2~~ ~~1~~ ~~0~~ 3 4 3 1 1 5 6 1 0

Página	Idade
3	1100
4	1000
1	0100
0	0100

Algoritmo LFU com envelhecimento

- > Memória com 4 molduras. Idade de 4 bits.
- > Limpeza a cada 4 acessos
- > Sequência: ~~3~~ ~~2~~ ~~1~~ ~~0~~ 3 4 3 1 1 5 6 1 0

Página	Idade
3	1100
4	1000
1	1100
0	0100

Algoritmo LFU com envelhecimento

- > Memória com 4 molduras. Idade de 4 bits.
- > Limpeza a cada 4 acessos
- > Sequência: ~~3~~ ~~2~~ ~~1~~ ~~0~~ ~~3~~ ~~4~~ ~~3~~ ~~1~~ 1 5 6 1 0

Página	Idade
3	0110
4	0100
1	0110
0	0010

Algoritmo LFU com envelhecimento

- > Memória com 4 molduras. Idade de 4 bits.
- > Limpeza a cada 4 acessos
- > Sequência: ~~3~~ ~~2~~ ~~1~~ ~~0~~ ~~3~~ ~~4~~ ~~3~~ ~~1~~ 1 5 6 1 0

Página	Idade
3	0110
4	0100
1	1110
0	0010

Algoritmo LFU com envelhecimento

- > Memória com 4 molduras. Idade de 4 bits.
- > Limpeza a cada 4 acessos
- > Sequência: ~~3~~ ~~2~~ ~~1~~ ~~0~~ ~~3~~ ~~4~~ ~~3~~ ~~1~~ 1 5 6 1 0

Página	Idade
3	0110
4	0100
1	1110
0	0010

Algoritmo LFU com envelhecimento

- > Memória com 4 molduras. Idade de 4 bits.
- > Limpeza a cada 4 acessos
- > Sequência: ~~3 2 1 0~~ ~~3 4 3 1~~ 1 5 6 1 0

Página	Idade
3	0110
4	0100
1	1110
5	1000

Algoritmo LFU com envelhecimento

- > Memória com 4 molduras. Idade de 4 bits.
- > Limpeza a cada 4 acessos
- > Sequência: ~~3~~ ~~2~~ ~~1~~ ~~0~~ ~~3~~ ~~4~~ ~~3~~ ~~1~~ 1 5 6 1 0

Página	Idade
3	0110
4	0100
1	1110
5	1000

Algoritmo LFU com envelhecimento

- > Memória com 4 molduras. Idade de 4 bits.
- > Limpeza a cada 4 acessos
- > Sequência: ~~3~~ ~~2~~ ~~1~~ ~~0~~ ~~3~~ ~~4~~ ~~3~~ ~~1~~ 1 5 6 1 0

Página	Idade
3	0110
6	1000
1	1110
5	1000

Algoritmo LFU com envelhecimento

- > Memória com 4 molduras. Idade de 4 bits.
- > Limpeza a cada 4 acessos
- > Sequência: ~~3 2 1 0~~ ~~3 4 3 1~~ 1 5 6 1 0

Página	Idade
3	0110
6	1000
1	1110
5	1000

Algoritmo LFU com envelhecimento

- > Memória com 4 molduras. Idade de 4 bits.
- > Limpeza a cada 4 acessos
- > Sequência: ~~3 2 1 0~~ ~~3 4 3 1~~ ~~1 5 6 1~~ 0

Página	Idade
3	0011
6	0100
1	0111
5	0100

Algoritmo LFU com envelhecimento

- > Memória com 4 molduras. Idade de 4 bits.
- > Limpeza a cada 4 acessos
- > Sequência: ~~3~~ ~~2~~ ~~1~~ ~~0~~ ~~3~~ ~~4~~ ~~3~~ ~~1~~ ~~1~~ ~~5~~ ~~6~~ ~~1~~ **0**

Página	Idade
3	0011
6	0100
1	0111
5	0100

Algoritmo LFU com envelhecimento

- > Memória com 4 molduras. Idade de 4 bits.
- > Limpeza a cada 4 acessos
- > Sequência: ~~3 2 1 0~~ ~~3 4 3 1~~ ~~1 5 6 1~~ 0

Página	Idade
0	1000
6	0100
1	0111
5	0100

Algoritmos globais x algoritmos locais

- > Globais: escolhem uma página de qualquer processo em memória principal para substituir
- > Locais: escolhem uma página do próprio processo em memória principal para substituir

Conjunto de trabalho (*working set*)

- > Conjunto mínimo de páginas necessário para o processo realizar o seu trabalho
- > Definição do conjunto de trabalho:
 - > Previsão do futuro
 - > Estimativa a partir de um intervalo de tempo passado
- > Proteção de uma página do conjunto
 - > Bit L (*lock*)

Algoritmo WSClock

- > Usa o conceito de conjunto de trabalho e o método de implementação do relógio.
- > Regras:
 - > Se $R=1$, zerar o bit R e seguir em frente
 - > Se $R=0$, verificar idade (tempo - intervalo)
 - Se idade válida, seguir em frente
 - Se idade inválida, verificar bit M
 - Se $M=0$, substituir página
 - Se $M=1$, marcar página e procurar outra candidata

Algoritmo WSClock

Tempo atual: 2204

Reset de R: 100

Intervalo WS: 200


4	0	1	1620
---	---	---	------

6	0	1	2084
---	---	---	------

5	1	1	2132
---	---	---	------

8	0	0	2003
---	---	---	------

3	1	0	2120
---	---	---	------



2	0	0	1213
---	---	---	------

7	0	0	2014
---	---	---	------

9	0	1	1980
---	---	---	------

Algoritmo WSClock

Tempo atual: 2204

Reset de R: 100

Intervalo WS: 200


4	0	1	1620
---	---	---	------

6	0	1	2084
---	---	---	------

5	1	1	2132
---	---	---	------

8	0	0	2003
---	---	---	------

3	0	0	2120
---	---	---	------



2	0	0	1213
---	---	---	------

7	0	0	2014
---	---	---	------

9	0	1	1980
---	---	---	------

Algoritmo WSClock

Tempo atual: 2204

Reset de R: 100

Intervalo WS: 200

4	0	1	1620
---	---	---	------

6	0	1	2084
---	---	---	------

5	1	1	2132
---	---	---	------

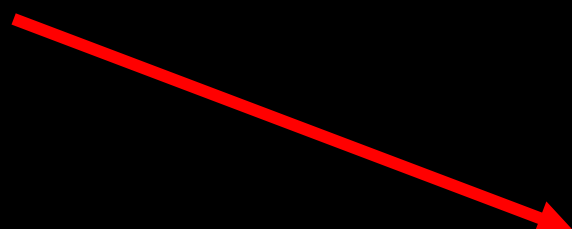
8	0	0	2003
---	---	---	------

3	0	0	2120
---	---	---	------

2	0	0	1213
---	---	---	------

7	0	0	2014
---	---	---	------

9	0	1	1980
---	---	---	------



Algoritmo WSClock

Tempo atual: 2204

Reset de R: 100

Intervalo WS: 200

4	0	1	1620
---	---	---	------

6	0	1	2084
---	---	---	------

5	1	1	2132
---	---	---	------

8	0	0	2003
---	---	---	------

3	0	0	2120
---	---	---	------

2	0	0	1213
---	---	---	------

7	0	0	2014
---	---	---	------

9	0	1	1980
---	---	---	------



Algoritmo WSClock

Tempo atual: 2204

Reset de R: 100

Intervalo WS: 200

4	0	1	1620
---	---	---	------

6	0	1	2084
---	---	---	------

5	1	1	2132
---	---	---	------

8	0	0	2003
---	---	---	------

3	0	0	2120
---	---	---	------

2	0	0	1213
---	---	---	------

7	0	0	2014
---	---	---	------

9	0	1	1980
---	---	---	------



Algoritmo WSClock

Tempo atual: 2204

Reset de R: 100

Intervalo WS: 200

4	0	1	1620
---	---	---	------

6	0	1	2084
---	---	---	------

5	1	1	2132
---	---	---	------

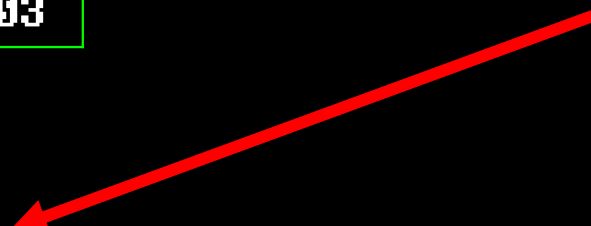
8	0	0	2003
---	---	---	------

3	0	0	2120
---	---	---	------

2	0	0	1213
---	---	---	------

7	0	0	2014
---	---	---	------

9	0	1	1980
---	---	---	------



Algoritmo WSClock

Tempo atual: 2204

Reset de R: 100

Intervalo WS: 200

4	0	1	1620
---	---	---	------

6	0	1	2084
---	---	---	------

5	1	1	2132
---	---	---	------

8	0	0	2003
---	---	---	------

3	0	0	2120
---	---	---	------

10	1	0	2204
----	---	---	------



7	0	0	2014
---	---	---	------

9	0	1	1980
---	---	---	------

Algoritmo WSClock

Tempo atual: 2205

Reset de R: 100

Intervalo WS: 200

4	0	1	1620
---	---	---	------

6	0	1	2084
---	---	---	------

5	1	1	2132
---	---	---	------

8	0	0	2003
---	---	---	------



3	0	0	2120
---	---	---	------

10	1	0	2204
----	---	---	------

7	0	0	2014
---	---	---	------

9	0	1	1980
---	---	---	------