Projeto 2: Sistemas Operacionais

3938 - Sayonara Bernardes
5144 - Douglas Boaventura
5164 - Milena Nobres

Algoritmos.

Para resolução desse trabalho foram implementados dois algoritmos sendo eles o algoritmo FIFO (first in first out) e o algoritmo de *second chance* ambos utilizando como base o código de um simulador de substituição de pagina disponibilizado pelo professor em sala.

FIFO

O algoritmo de FIFO utilizado não somente em substituição de pagina mais também em várias outras áreas é um algoritmo de execução e implementação bem simples. O primeiro elemento que entrou na fila é o primeiro a sair, ou seja, se uma página X deve entrar nos *frames* quem deve ceder seu lugar então é a página que está a mais tempo na paginação física, o algoritmo tem uma implementação bem simples e por sua vez foi implementado da seguinte forma:

```
// fifo_frm    o frame de inicio da fila, ent o    feito um loop
// buscando na tabela a pagina com esse valor e assim retornando
// como pagina a ser substituida
for (int i = 0; i < num_pages; i++) {
    if (page_table[i][PT_FRAMEID] == fifo_frm) return i;
}
```

SECOND CHANCE

O algoritmo de *second chance* segue um principio extremamente próximo ao do *fifo*, com uma única exceção, ele tem o bit R de referencia responsável por ditar se o *frame* tem uma segunda chance.

Assim como anterior analisa-se primeiramente a cabeça da fila, se o elemento não tem o bit de segunda chance o procedimento é o mesmo do FIFO, mas, caso o bit de segunda chance seja 1 o elemento é removido do inicio e inserido no final, mas perde seu bit de segunda chance, a cabeça da fila é verificada e o procedimento se inicia novamente.

Caso no meio do processo algum *frame* com bit R de 0 seja referenciado, seu bit R volta a valer 1, a seguir vemos como foi implementado esse algoritmo no código:

```
// primeiro loop onde j percorre o numero de molduras pois a fila
// deve ser alterada caso exista uma segunda chance no primeiro da fila
for (int j = fifo_frm; 1; j = (j + 1) % num_frames) {
    // i percorre as paginas para procurar quem pertence ao frame j
  for (int i = 0; i < num_pages; i++) {
    if (page_table[i][PT_FRAMEID] == j) {
      if (page_table[i][PT_REFERENCE_BIT] == 0) {
                // caso o bit de referencia    0 ele n tem segunda
                // chance, ent o    retornado como elemento a ser
                // substituido do frame.
        return i;
      } else {
                // se o bit de referencia    1 ele    decrementado e o
                // loop passa para o proximo elemento da fila
                // utilizando (j + 1) % num_frame
        page_table[i][PT_REFERENCE_BIT] = 0;
      }
    }
  }
}
```

RANDOM

Algoritmo fornecido pelo professor e tem como base ser totalmente aleatório em qual casa ele substitui, segue a sua implementação:

```
int page = rand() % num_pages;
while (page_table[page][PT_MAPPED] == 0) // Encontra p gina mapeada
    page = rand() % num_pages;
return page;
```

Resultados

Os resultados estão distribuídos na tabela a seguir após testado em 10 entradas de teste diferentes dada pelos arquivos *anomaly* anexados ao código:

Tabela 1: Tabela de resultados valor de erros nos algoritmos por entrada de dados.

Entrada	FIFO	SECOND CHANCE	RANDOM
1	9	9	9
2	10	9	11
3	27	23	26
4	10	10	7
5	7	7	9
6	9	9	12
7	12	12	12
8	14	15	16
9	13	13	13
10	12	12	13