Sistemas Operativos

Tarea 2
Informe

Integrantes:
Ignacio F. Garcés
Francisco Jiménez

1. INSTRUCTIVO DE USO DEL PROGRAMA

Al ejecutar `make`, se generará el ejecutable `virtmem`. Luego, la forma de ejecutar `virtmem` es

```
./virtmem <N° páginas> <N° marcos> <algoritmo cambio de página> <patrón de acceso memoria>
```

Es decir,

```
./virtmem <npages> <nframes> fifo|rand seq|rand|rev
```

Ejemplo:

```
./virtmem 50 20 fifo seq
```

Los patrones de acceso a memoria pueden ser secuencial (`seq`), aleatorio (`rand`) o secuencial inverso (`rev`).

2. COMPARACIÓN DE POLÍTICAS DE REEMPLAZO DE PÁGINA

Comparando usando patrón de acceso de memoria secuencial:

2.1. FIFO

Con 80 páginas y 50 marcos, ocurren 190 faltas de página y 110 reemplazos.

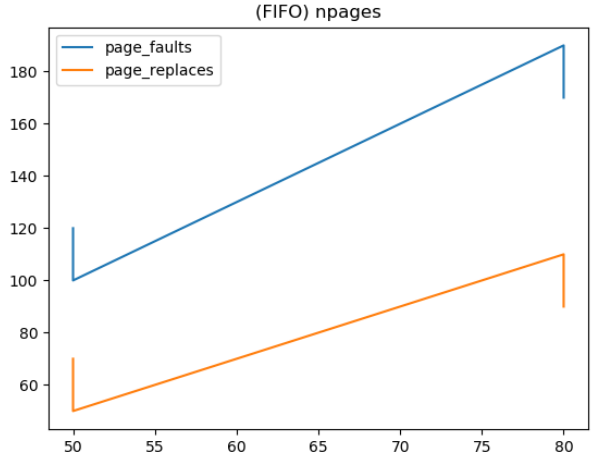
Input:

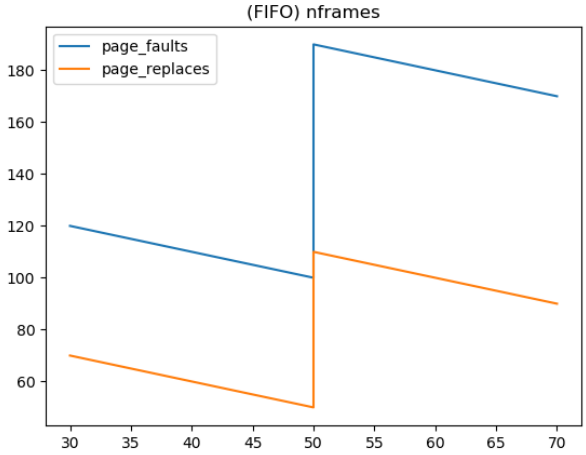
```
./virtmem 80 50 fifo seq
```

Output:

```
page fault on page #0
...
page fault on page #79
Cantidad de faltas de página: 190
Cantidad de reemplazos de página 110
```

Graficando con ayuda de Python (`graphthis.py`) con datos de input-output probados:





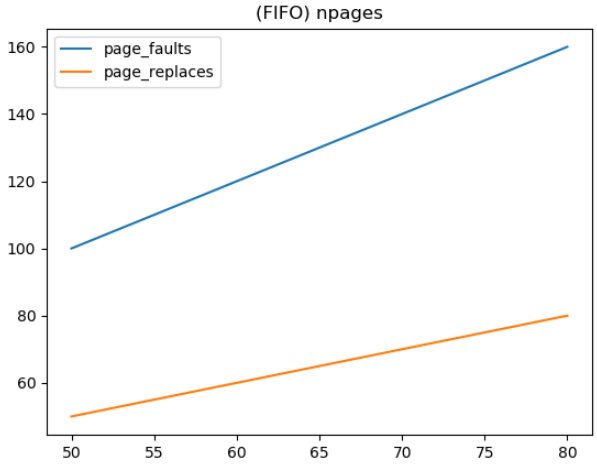
Fueron probados los inputs:

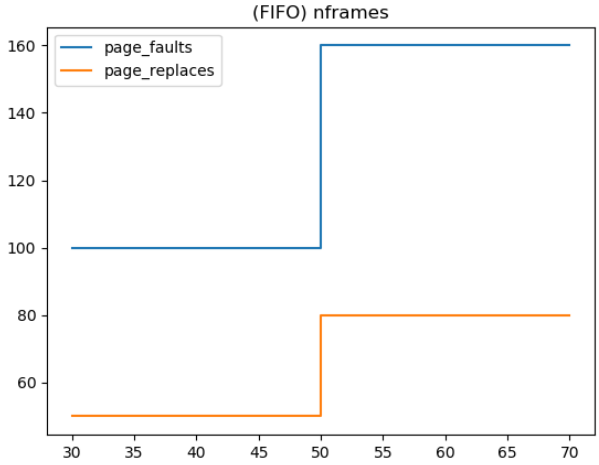
```
./virtmem 50 30 fifo seq
./virtmem 50 50 fifo seq
./virtmem 80 50 fifo seq
./virtmem 80 70 fifo seq
```

Se ve que la cantidad de reemplazo de páginas y faltas de páginas son proporcionales entre sí.

2.2. ALEATORIO

(los títulos de los gráficos deben decir Random en vez de FIFO)





Se usaron los mismos input que en el caso de FIFO, y sin embargo dio una eficiencia mayor.