

Министерство науки и высшего образования
Российской Федерации Федеральное
государственное автономное образовательное
учреждение высшего образования «Национальный
исследовательский университет ИТМО»

Факультет Программной Инженерии и
Компьютерной Техники

Операционные системы
Лабораторная работа №1. Вариант 91582

Выполнил студент:	Соловьев Павел Андреевич
Группа:	Р33302
Преподаватель:	Макарьев Евгений Юрьевич

Санкт-Петербург
2024

Задание

Задание 2. Программа работает в операционной системе, которая осуществляет замещение кадров основной памяти страницами во вторичной памяти. При обращении к странице, которая отсутствует в основной памяти, происходит замещение страницы по заданному алгоритму.

Количество кадров в основной памяти, выделенных программе, равно 7.

Кадры в основной памяти в начале работы программы не инициализированы.

Количество страниц в виртуальной памяти процесса равно 24.

Программа осуществляет обращения к страницам в следующем порядке:

[7, 6, 4, 12, 20, 3, 8, 11, 13, 6, 1, 8, 24, 11, 16, 10, 14, 7, 13, 8, 23, 8, 22, 21, 11, 18, 22, 23, 22, 12, 21, 6, 17, 18, 20, 19, 9]

Рассмотреть стратегии замещения - Оптимальную, LRU, FIFO (Столлингс, гл.8.2). Для каждого алгоритма:

- нарисовать состояние кадров основной памяти во время обращения программы;
- определить количество операций по замене страниц;
- сравнить количество замен по сравнению с оптимальным.

Как изменится количество замен страниц, если увеличить количество кадров в 2 раза?

А если уменьшить количество кадров в 2 раза?

Сколько должно кадров в памяти, чтобы оптимальный алгоритм давал 5% страничных сбоев?

Ход работы

Была написана программа на Go, рассматривающая данные стратегии, также учитывающая уменьшение и увеличение количества кадров.

```
package page_replacement

import (
    "fmt"
    "slices"
)

type BasicPageReplacerWrapper struct {
    Replacer      Replacer
    AccessNotifier AccessNotifier
    frames         []int
    framesCount    int
    totalPages     int
    pagesAccesses []int

    pageFaults int
}

type Replacer interface {
    ChoosePageIdxToReplace(currentIndex int, pagesAccesses []int, frames []int) int
}

type AccessNotifier interface {
    Notify(page int, currentIndex int)
}

func NewBasicPageReplacerWrapper(replacer Replacer, framesCount, totalPages int, pagesAccesses []int, notifier AccessNotifier) *BasicPageReplacerWrapper {
    frames := make([]int, framesCount)
    for i := 0; i < framesCount; i++ {
        frames[i] = -1
    }
    return &BasicPageReplacerWrapper{
        Replacer:      replacer,
        frames:         frames,
        framesCount:    framesCount,
        totalPages:     totalPages,
        pagesAccesses:  pagesAccesses,
        AccessNotifier: notifier,
    }
}

func (b *BasicPageReplacerWrapper) Run(verbose bool, isEmptyPageFault bool) {
    if verbose {
        b.printHeading()
    }
    for i := 0; i < len(b.pagesAccesses); i++ {
        pageToAccess := b.pagesAccesses[i]
        b.AccessNotifier.Notify(pageToAccess, i)
        isFault := !b.isPageInFrames(pageToAccess)
        isEmpty := slices.Contains(b.frames, -1)
```

```

isShowFault := (!isEmpty || isEmptyPageFault) && isFault
if isFault {
    pageIndex := getFreeFrame(b.frames)
    if pageIndex == -1 {
        pageIndex = b.Replacer.ChoosePageIdxToReplace(i, b.pagesAccesses, b.frames)
    }
    b.frames[pageIndex] = pageToAccess
    if isShowFault {
        b.pageFaults++
    }
}
if !verbose {
    continue
}
b.Print(pageToAccess, isShowFault)
}
}

func (b *BasicPageReplacerWrapper) GetPageFaults() int {
    return b.pageFaults
}

func getFreeFrame(frames []int) int {
    for i := 0; i < len(frames); i++ {
        if frames[i] == -1 {
            return i
        }
    }
    return -1
}

func (b *BasicPageReplacerWrapper) isPageInFrames(page int) bool {
    for i := 0; i < b.framesCount; i++ {
        if b.frames[i] == page {
            return true
        }
    }
    return false
}

func (b *BasicPageReplacerWrapper) printHeading() {
    fmt.Print(" P | ")
    for i := 0; i < b.framesCount; i++ {
        fmt.Printf("f%d ", i+1)
    }
    fmt.Print(" | fault?")
    fmt.Println()
    fmt.Println("----+-----+-----")
}

func (b *BasicPageReplacerWrapper) Print(pageToAccess int, isFault bool) {
    fmt.Printf("%2d | ", pageToAccess)
    for i := 0; i < b.framesCount; i++ {
        fmt.Printf("%2d ", b.frames[i])
    }
    fmt.Print(" | ")
    if isFault {
        fmt.Print("Page fault")
    }
    fmt.Println()
}

type FIFO struct {
    framesCount int
    indexToReplace int
}

func NewFIFO(framesCount int) Replacer {
    return &FIFO{
        framesCount: framesCount,
        indexToReplace: 0,
    }
}

func (f *FIFO) ChoosePageIdxToReplace(int, []int, []int) int {
    lastIndex := f.indexToReplace
    f.indexToReplace++
    if f.indexToReplace == f.framesCount {
        f.indexToReplace = 0
    }
    return lastIndex
}

```

```

}

type LRU struct {
    lastAccessesPageToTime map[int]int
}

func NewLRU(totalPagesCount int) *LRU {
    return &LRU{
        lastAccessesPageToTime: make(map[int]int, totalPagesCount),
    }
}

func (l *LRU) Notify(page int, currentIndex int) {
    l.lastAccessesPageToTime[page] = currentIndex
}

func (l *LRU) ChoosePageIdxToReplace(currentIndex int, pagesAccesses []int, frames []int) int {
    minTime := currentIndex
    pageValue := -1
    for i := 0; i < len(pagesAccesses); i++ {
        page := pagesAccesses[i]
        if l.lastAccessesPageToTime[page] < minTime && slices.Contains(frames, page) {
            minTime = l.lastAccessesPageToTime[page]
            pageValue = page
        }
    }
    for i := 0; i < len(frames); i++ {
        if frames[i] == pageValue {
            return i
        }
    }
    panic(fmt.Sprintf("Page %d not found in frames", pageValue))
}

type NoopNotifier struct{}

func (n *NoopNotifier) Notify(int, int) {}

type OPT struct {
    totalPages int
}

func NewOPT(totalPages int) *OPT {
    return &OPT{
        totalPages: totalPages,
    }
}

func (o *OPT) ChoosePageIdxToReplace(currentIndex int, pagesAccesses []int, frames []int) int {
    maxDistance := 0
    pageValue := frames[0]
    for i := 0; i < len(pagesAccesses); i++ {
        page := pagesAccesses[i]
        if o.distanceToNextReference(page, currentIndex, pagesAccesses) > maxDistance && slices.Contains(frames, page) {
            maxDistance = o.distanceToNextReference(page, currentIndex, pagesAccesses)
            pageValue = page
        }
    }
    for i := 0; i < len(frames); i++ {
        if frames[i] == pageValue {
            return i
        }
    }
    panic(fmt.Sprintf("Page %d not found in frames", pageValue))
}

func (o *OPT) distanceToNextReference(page, currentIndex int, pagesAccesses []int) int {
    for i := currentIndex; i < len(pagesAccesses); i++ {
        if pagesAccesses[i] == page {
            return i - currentIndex
        }
    }
    return o.totalPages
}

```

Вывод программы

Using 'fifo' page replacement algorithm

P | f1 f2 f3 f4 f5 f6 f7 | fault?

```

-----+-----
7 | 7 -1 -1 -1 -1 -1 -1 |

```

6	7	6	-1	-1	-1	-1	-1	
4	7	6	4	-1	-1	-1	-1	
12	7	6	4	12	-1	-1	-1	
20	7	6	4	12	20	-1	-1	
3	7	6	4	12	20	3	-1	
8	7	6	4	12	20	3	8	
11	11	6	4	12	20	3	8	Page fault
13	11	13	4	12	20	3	8	Page fault
6	11	13	6	12	20	3	8	Page fault
1	11	13	6	1	20	3	8	Page fault
8	11	13	6	1	20	3	8	
24	11	13	6	1	24	3	8	Page fault
11	11	13	6	1	24	3	8	
6	11	13	6	1	24	3	8	
10	11	13	6	1	24	10	8	Page fault
14	11	13	6	1	24	10	14	Page fault
7	7	13	6	1	24	10	14	Page fault
13	7	13	6	1	24	10	14	
8	7	8	6	1	24	10	14	Page fault
23	7	8	23	1	24	10	14	Page fault
8	7	8	23	1	24	10	14	
22	7	8	23	22	24	10	14	Page fault
21	7	8	23	22	21	10	14	Page fault
11	7	8	23	22	21	11	14	Page fault
18	7	8	23	22	21	11	18	Page fault
22	7	8	23	22	21	11	18	
23	7	8	23	22	21	11	18	
22	7	8	23	22	21	11	18	
12	12	8	23	22	21	11	18	Page fault
21	12	8	23	22	21	11	18	
6	12	6	23	22	21	11	18	Page fault
17	12	6	17	22	21	11	18	Page fault
18	12	6	17	22	21	11	18	
20	12	6	17	20	21	11	18	Page fault
19	12	6	17	20	19	11	18	Page fault
9	12	6	17	20	19	9	18	Page fault

Statistics:

faults/non-faults/total accesses: 20/17/37

Total page faults 20. Page faults with optimal algo: 14

Using 'lru' page replacement algorithm

P	f1	f2	f3	f4	f5	f6	f7	fault?

7	7	-1	-1	-1	-1	-1	-1	
6	7	6	-1	-1	-1	-1	-1	
4	7	6	4	-1	-1	-1	-1	
12	7	6	4	12	-1	-1	-1	
20	7	6	4	12	20	-1	-1	
3	7	6	4	12	20	3	-1	
8	7	6	4	12	20	3	8	
11	11	6	4	12	20	3	8	Page fault
13	11	13	4	12	20	3	8	Page fault
6	11	13	6	12	20	3	8	Page fault
1	11	13	6	1	20	3	8	Page fault
8	11	13	6	1	20	3	8	
24	11	13	6	1	24	3	8	Page fault
11	11	13	6	1	24	3	8	
6	11	13	6	1	24	3	8	
10	11	13	6	1	24	10	8	Page fault
14	11	14	6	1	24	10	8	Page fault
7	11	14	6	7	24	10	8	Page fault
13	11	14	6	7	24	10	13	Page fault
8	11	14	6	7	8	10	13	Page fault
23	23	14	6	7	8	10	13	Page fault
8	23	14	6	7	8	10	13	
22	23	14	22	7	8	10	13	Page fault
21	23	14	22	7	8	21	13	Page fault
11	23	11	22	7	8	21	13	Page fault
18	23	11	22	18	8	21	13	Page fault
22	23	11	22	18	8	21	13	
23	23	11	22	18	8	21	13	
22	23	11	22	18	8	21	13	
12	23	11	22	18	8	21	12	Page fault
21	23	11	22	18	8	21	12	
6	23	11	22	18	6	21	12	Page fault
17	23	17	22	18	6	21	12	Page fault
18	23	17	22	18	6	21	12	
20	20	17	22	18	6	21	12	Page fault
19	20	17	19	18	6	21	12	Page fault
9	20	17	19	18	6	21	9	Page fault

Statistics:

faults/non-faults/total accesses: 21/16/37

Total page faults 21. Page faults with optimal algo: 14

Using 'opt' page replacement algorithm

P	f1	f2	f3	f4	f5	f6	f7	fault?
7	7	-1	-1	-1	-1	-1	-1	
6	7	6	-1	-1	-1	-1	-1	
4	7	6	4	-1	-1	-1	-1	
12	7	6	4	12	-1	-1	-1	
20	7	6	4	12	20	-1	-1	
3	7	6	4	12	20	3	-1	
8	7	6	4	12	20	3	8	
11	7	6	4	12	11	3	8	Page fault
13	7	6	13	12	11	3	8	Page fault
6	7	6	13	12	11	3	8	
1	7	6	13	12	11	1	8	Page fault
8	7	6	13	12	11	1	8	
24	7	6	13	12	11	24	8	Page fault
11	7	6	13	12	11	24	8	
6	7	6	13	12	11	24	8	
10	7	6	13	12	11	10	8	Page fault
14	7	6	13	12	11	14	8	Page fault
7	7	6	13	12	11	14	8	
13	7	6	13	12	11	14	8	
8	7	6	13	12	11	14	8	
23	23	6	13	12	11	14	8	Page fault
8	23	6	13	12	11	14	8	
22	23	6	13	12	11	14	22	Page fault
21	23	6	21	12	11	14	22	Page fault
11	23	6	21	12	11	14	22	
18	23	6	21	12	18	14	22	Page fault
22	23	6	21	12	18	14	22	
23	23	6	21	12	18	14	22	
22	23	6	21	12	18	14	22	
12	23	6	21	12	18	14	22	
21	23	6	21	12	18	14	22	
6	23	6	21	12	18	14	22	
17	23	17	21	12	18	14	22	Page fault
18	23	17	21	12	18	14	22	
20	23	17	21	20	18	14	22	Page fault
19	23	17	21	19	18	14	22	Page fault
9	23	17	21	19	18	9	22	Page fault

Statistics:

faults/non-faults/total accesses: 14/23/37

Total page faults 14. Page faults with optimal algo: 14

Количество кадров: 3

Using 'fifo' page replacement algorithm

P	f1	f2	f3	fault?
7	7	-1	-1	
6	7	6	-1	
4	7	6	4	
12	12	6	4	Page fault
20	12	20	4	Page fault
3	12	20	3	Page fault
8	8	20	3	Page fault
11	8	11	3	Page fault
13	8	11	13	Page fault
6	6	11	13	Page fault
1	6	1	13	Page fault
8	6	1	8	Page fault
24	24	1	8	Page fault
11	24	11	8	Page fault
6	24	11	6	Page fault
10	10	11	6	Page fault
14	10	14	6	Page fault
7	10	14	7	Page fault
13	13	14	7	Page fault
8	13	8	7	Page fault
23	13	8	23	Page fault
8	13	8	23	
22	22	8	23	Page fault
21	22	21	23	Page fault
11	22	21	11	Page fault
18	18	21	11	Page fault
22	18	22	11	Page fault

23	18	22	23	Page fault
22	18	22	23	
12	12	22	23	Page fault
21	12	21	23	Page fault
6	12	21	6	Page fault
17	17	21	6	Page fault
18	17	18	6	Page fault
20	17	18	20	Page fault
19	19	18	20	Page fault
9	19	9	20	Page fault

Statistics:

faults/non-faults/total accesses: 32/5/37

Total page faults 32. Page faults with optimal algo: 25

Using 'lru' page replacement algorithm

P	f1	f2	f3	fault?
7	7	-1	-1	
6	7	6	-1	
4	7	6	4	
12	12	6	4	Page fault
20	12	20	4	Page fault
3	12	20	3	Page fault
8	8	20	3	Page fault
11	8	11	3	Page fault
13	8	11	13	Page fault
6	6	11	13	Page fault
1	6	1	13	Page fault
8	6	1	8	Page fault
24	24	1	8	Page fault
11	24	11	8	Page fault
6	24	11	6	Page fault
10	10	11	6	Page fault
14	10	14	6	Page fault
7	10	14	7	Page fault
13	13	14	7	Page fault
8	13	8	7	Page fault
23	13	8	23	Page fault
8	13	8	23	
22	22	8	23	Page fault
21	22	8	21	Page fault
11	22	11	21	Page fault
18	18	11	21	Page fault
22	18	11	22	Page fault
23	18	23	22	Page fault
22	18	23	22	
12	12	23	22	Page fault
21	12	21	22	Page fault
6	12	21	6	Page fault
17	17	21	6	Page fault
18	17	18	6	Page fault
20	17	18	20	Page fault
19	19	18	20	Page fault
9	19	9	20	Page fault

Statistics:

faults/non-faults/total accesses: 32/5/37

Total page faults 32. Page faults with optimal algo: 25

Using 'opt' page replacement algorithm

P	f1	f2	f3	fault?
7	7	-1	-1	
6	7	6	-1	
4	7	6	4	
12	7	6	12	Page fault
20	7	6	20	Page fault
3	7	6	3	Page fault
8	7	6	8	Page fault
11	11	6	8	Page fault
13	13	6	8	Page fault
6	13	6	8	
1	1	6	8	Page fault
8	1	6	8	
24	24	6	8	Page fault
11	11	6	8	Page fault
6	11	6	8	
10	11	10	8	Page fault
14	11	14	8	Page fault

7		11	7	8		Page fault
13		11	13	8		Page fault
8		11	13	8		
23		11	23	8		Page fault
8		11	23	8		
22		11	23	22		Page fault
21		11	21	22		Page fault
11		11	21	22		
18		18	21	22		Page fault
22		18	21	22		
23		23	21	22		Page fault
22		23	21	22		
12		12	21	22		Page fault
21		12	21	22		
6		6	21	22		Page fault
17		17	21	22		Page fault
18		17	21	18		Page fault
20		17	20	18		Page fault
19		17	19	18		Page fault
9		17	19	9		Page fault

Statistics:

faults/non-faults/total accesses: 25/12/37

Total page faults 25. Page faults with optimal algo: 25

Количество кадров: 14

Using 'fifo' page replacement algorithm

P		f1	f2	f3	f4	f5	f6	f7	f8	f9	f10	f11	f12	f13	f14		fault?
-----+-----																	
7		7	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1		
6		7	6	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1		
4		7	6	4	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1		
12		7	6	4	12	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1		
20		7	6	4	12	20	-1	-1	-1	-1	-1	-1	-1	-1	-1		
3		7	6	4	12	20	3	-1	-1	-1	-1	-1	-1	-1	-1		
8		7	6	4	12	20	3	8	-1	-1	-1	-1	-1	-1	-1		
11		7	6	4	12	20	3	8	11	-1	-1	-1	-1	-1	-1		
13		7	6	4	12	20	3	8	11	13	-1	-1	-1	-1	-1		
6		7	6	4	12	20	3	8	11	13	-1	-1	-1	-1	-1		
1		7	6	4	12	20	3	8	11	13	1	-1	-1	-1	-1		
8		7	6	4	12	20	3	8	11	13	1	-1	-1	-1	-1		
24		7	6	4	12	20	3	8	11	13	1	24	-1	-1	-1		
11		7	6	4	12	20	3	8	11	13	1	24	-1	-1	-1		
6		7	6	4	12	20	3	8	11	13	1	24	-1	-1	-1		
10		7	6	4	12	20	3	8	11	13	1	24	10	-1	-1		
14		7	6	4	12	20	3	8	11	13	1	24	10	14	-1		
7		7	6	4	12	20	3	8	11	13	1	24	10	14	-1		
13		7	6	4	12	20	3	8	11	13	1	24	10	14	-1		
8		7	6	4	12	20	3	8	11	13	1	24	10	14	-1		
23		7	6	4	12	20	3	8	11	13	1	24	10	14	23		
8		7	6	4	12	20	3	8	11	13	1	24	10	14	23		
22		22	6	4	12	20	3	8	11	13	1	24	10	14	23		Page fault
21		22	21	4	12	20	3	8	11	13	1	24	10	14	23		Page fault
11		22	21	4	12	20	3	8	11	13	1	24	10	14	23		
18		22	21	18	12	20	3	8	11	13	1	24	10	14	23		Page fault
22		22	21	18	12	20	3	8	11	13	1	24	10	14	23		
23		22	21	18	12	20	3	8	11	13	1	24	10	14	23		
22		22	21	18	12	20	3	8	11	13	1	24	10	14	23		
12		22	21	18	12	20	3	8	11	13	1	24	10	14	23		
21		22	21	18	12	20	3	8	11	13	1	24	10	14	23		
6		22	21	18	6	20	3	8	11	13	1	24	10	14	23		Page fault
17		22	21	18	6	17	3	8	11	13	1	24	10	14	23		Page fault
18		22	21	18	6	17	3	8	11	13	1	24	10	14	23		
20		22	21	18	6	17	20	8	11	13	1	24	10	14	23		Page fault
19		22	21	18	6	17	20	19	11	13	1	24	10	14	23		Page fault
9		22	21	18	6	17	20	19	9	13	1	24	10	14	23		Page fault

Statistics:

faults/non-faults/total accesses: 8/29/37

Total page faults 8. Page faults with optimal algo: 6

Using 'lru' page replacement algorithm

P		f1	f2	f3	f4	f5	f6	f7	f8	f9	f10	f11	f12	f13	f14		fault?
-----+-----																	
7		7	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1		
6		7	6	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1		
4		7	6	4	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1		
12		7	6	4	12	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1		
20		7	6	4	12	20	-1	-1	-1	-1	-1	-1	-1	-1	-1		
3		7	6	4	12	20	3	-1	-1	-1	-1	-1	-1	-1	-1		
8		7	6	4	12	20	3	8	-1	-1	-1	-1	-1	-1	-1		

11	7	6	4	12	20	3	8	11	-1	-1	-1	-1	-1	-1
13	7	6	4	12	20	3	8	11	13	-1	-1	-1	-1	-1
6	7	6	4	12	20	3	8	11	13	-1	-1	-1	-1	-1
1	7	6	4	12	20	3	8	11	13	1	-1	-1	-1	-1
8	7	6	4	12	20	3	8	11	13	1	-1	-1	-1	-1
24	7	6	4	12	20	3	8	11	13	1	24	-1	-1	-1
11	7	6	4	12	20	3	8	11	13	1	24	-1	-1	-1
6	7	6	4	12	20	3	8	11	13	1	24	-1	-1	-1
10	7	6	4	12	20	3	8	11	13	1	24	10	-1	-1
14	7	6	4	12	20	3	8	11	13	1	24	10	14	-1
7	7	6	4	12	20	3	8	11	13	1	24	10	14	-1
13	7	6	4	12	20	3	8	11	13	1	24	10	14	-1
8	7	6	4	12	20	3	8	11	13	1	24	10	14	-1
23	7	6	4	12	20	3	8	11	13	1	24	10	14	23
8	7	6	4	12	20	3	8	11	13	1	24	10	14	23
22	7	6	22	12	20	3	8	11	13	1	24	10	14	23
21	7	6	22	21	20	3	8	11	13	1	24	10	14	23
11	7	6	22	21	20	3	8	11	13	1	24	10	14	23
18	7	6	22	21	18	3	8	11	13	1	24	10	14	23
22	7	6	22	21	18	3	8	11	13	1	24	10	14	23
23	7	6	22	21	18	3	8	11	13	1	24	10	14	23
22	7	6	22	21	18	3	8	11	13	1	24	10	14	23
12	7	6	22	21	18	12	8	11	13	1	24	10	14	23
21	7	6	22	21	18	12	8	11	13	1	24	10	14	23
6	7	6	22	21	18	12	8	11	13	1	24	10	14	23
17	7	6	22	21	18	12	8	11	13	17	24	10	14	23
18	7	6	22	21	18	12	8	11	13	17	24	10	14	23
20	7	6	22	21	18	12	8	11	13	17	20	10	14	23
19	7	6	22	21	18	12	8	11	13	17	20	19	14	23
9	7	6	22	21	18	12	8	11	13	17	20	19	9	23

Statistics:

faults/non-faults/total accesses: 8/29/37

Total page faults 8. Page faults with optimal algo: 6

Using 'opt' page replacement algorithm

P	f1	f2	f3	f4	f5	f6	f7	f8	f9	f10	f11	f12	f13	f14	fault?
7	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	
6	7	6	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	
4	7	6	4	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	
12	7	6	4	12	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	
20	7	6	4	12	20	-1	-1	-1	-1	-1	-1	-1	-1	-1	
3	7	6	4	12	20	3	-1	-1	-1	-1	-1	-1	-1	-1	
8	7	6	4	12	20	3	8	-1	-1	-1	-1	-1	-1	-1	
11	7	6	4	12	20	3	8	11	-1	-1	-1	-1	-1	-1	
13	7	6	4	12	20	3	8	11	13	-1	-1	-1	-1	-1	
6	7	6	4	12	20	3	8	11	13	-1	-1	-1	-1	-1	
1	7	6	4	12	20	3	8	11	13	1	-1	-1	-1	-1	
8	7	6	4	12	20	3	8	11	13	1	-1	-1	-1	-1	
24	7	6	4	12	20	3	8	11	13	1	24	-1	-1	-1	
11	7	6	4	12	20	3	8	11	13	1	24	-1	-1	-1	
6	7	6	4	12	20	3	8	11	13	1	24	-1	-1	-1	
10	7	6	4	12	20	3	8	11	13	1	24	10	-1	-1	
14	7	6	4	12	20	3	8	11	13	1	24	10	14	-1	
7	7	6	4	12	20	3	8	11	13	1	24	10	14	-1	
13	7	6	4	12	20	3	8	11	13	1	24	10	14	-1	
8	7	6	4	12	20	3	8	11	13	1	24	10	14	-1	
23	7	6	4	12	20	3	8	11	13	1	24	10	14	23	
8	7	6	4	12	20	3	8	11	13	1	24	10	14	23	
22	22	6	4	12	20	3	8	11	13	1	24	10	14	23	Page fault
21	22	6	21	12	20	3	8	11	13	1	24	10	14	23	Page fault
11	22	6	21	12	20	3	8	11	13	1	24	10	14	23	
18	22	6	21	12	20	18	8	11	13	1	24	10	14	23	Page fault
22	22	6	21	12	20	18	8	11	13	1	24	10	14	23	
23	22	6	21	12	20	18	8	11	13	1	24	10	14	23	
22	22	6	21	12	20	18	8	11	13	1	24	10	14	23	
12	22	6	21	12	20	18	8	11	13	1	24	10	14	23	
21	22	6	21	12	20	18	8	11	13	1	24	10	14	23	
6	22	6	21	12	20	18	8	11	13	1	24	10	14	23	
17	22	17	21	12	20	18	8	11	13	1	24	10	14	23	Page fault
18	22	17	21	12	20	18	8	11	13	1	24	10	14	23	
20	22	17	21	12	20	18	8	11	13	1	24	10	14	23	
19	22	17	21	19	20	18	8	11	13	1	24	10	14	23	Page fault
9	22	17	21	19	9	18	8	11	13	1	24	10	14	23	Page fault

Statistics:

faults/non-faults/total accesses: 6/31/37

Total page faults 6. Page faults with optimal algo: 6

5% страничных сбоев при оптимальном алгоритме

Frames: 1. Page faults percentage: 0.972973. (36/37)
Frames: 2. Page faults percentage: 0.810811. (30/37)
Frames: 3. Page faults percentage: 0.675676. (25/37)
Frames: 4. Page faults percentage: 0.567568. (21/37)
Frames: 5. Page faults percentage: 0.486486. (18/37)
Frames: 6. Page faults percentage: 0.432432. (16/37)
Frames: 7. Page faults percentage: 0.378378. (14/37)
Frames: 8. Page faults percentage: 0.351351. (13/37)
Frames: 9. Page faults percentage: 0.324324. (12/37)
Frames: 10. Page faults percentage: 0.270270. (10/37)
Frames: 11. Page faults percentage: 0.243243. (9/37)
Frames: 12. Page faults percentage: 0.216216. (8/37)
Frames: 13. Page faults percentage: 0.189189. (7/37)
Frames: 14. Page faults percentage: 0.162162. (6/37)
Frames: 15. Page faults percentage: 0.135135. (5/37)
Frames: 16. Page faults percentage: 0.108108. (4/37)
Frames: 17. Page faults percentage: 0.081081. (3/37)
Frames: 18. Page faults percentage: 0.054054. (2/37)
Frames: 19. Page faults percentage: 0.027027. (1/37)

Ответы на вопросы

- Как изменится количество замен страниц, если увеличить количество кадров в 2 раза?
уменьшится примерно в 2.5 раза
- А если уменьшить количество кадров в 2 раза? увеличится примерно в 1.6 раза
- Сколько должно кадров в памяти, чтобы оптимальный алгоритм давал 5% страничных сбоев? 19 и более