Exercise 8

Student's Name and ID: s1240078 Tomonori Masubuchi

Task 1

Suppose we have a demand-paged memory. The page table is held in registers. It takes 8 milliseconds to service a page fault if an empty page is available or the replaced page is not modified, and 20 milliseconds if the replaced page is modified. Memory access time is 100 nanoseconds.
Assume that the page to be replaced is modified 70 percent of the time.

Question: What is the maximum acceptable page-fault rate for an effective access time of no more than 200 nanoseconds?

(put your answer here)
200 = (1 - p) * 100 + 0.7 * p * 20000000 + 0.3 * p * 8000000
p = 0.061%

Task 2

Consider the following page reference string:
1, 2, 3, 4, 2, 1, 5, 6, 2, 1, 2, 3, 7, 6, 3, 2, 1, 2, 3, 6

How many page faults would occur for the following replacement algorithms, assuming three and four frames ? Remember all frames are initially empty, so your first unique pages will all cost one fault each.

1. LRU replacement
2. FIFO replacement
3. Optimal replacement

(fill tables to answer)

LRU; frame = 3																				
LRU	**1**	**2**	**3**	**4**	**2**	**1**	**5**	**6**	**2**	**1**	**2**	**3**	**7**	**6**	**3**	**2**	**1**	**2**	**3**	**6**
	1	1	1	4	4	4	5	5	5	1	1	1	7	7	7	2	2	2	2	2
		2	2	2	2	2	2	6	6	6	6	3	3	3	3	3	3	3	3	3
			3	3	3	1	1	1	2	2	2	2	2	6	6	6	1	1	1	6
Page Fault	✓	✓	✓	✓		✓	✓	✓	✓	✓		✓	✓	✓		✓	✓			✓

FIFO; frame = 3																				
FIFO	**1**	**2**	**3**	**4**	**2**	**1**	**5**	**6**	**2**	**1**	**2**	**3**	**7**	**6**	**3**	**2**	**1**	**2**	**3**	**6**
	1	1	1	4	4	4	4	6	6	6	6	3	3	3	3	2	2	2	2	6
		2	2	2	2	1	1	1	2	2	2	2	7	7	7	7	1	1	1	1
			3	3	3	3	5	5	5	1	1	1	1	6	6	6	6	6	3	3
Page Fault	✓	✓	✓	✓		✓	✓	✓	✓	✓		✓	✓	✓		✓	✓		✓	✓

OPT; frame = 3																				
OPT	**1**	**2**	**3**	**4**	**2**	**1**	**5**	**6**	**2**	**1**	**2**	**3**	**7**	**6**	**3**	**2**	**1**	**2**	**3**	**6**
	1	1	1	4	4	4	5	5	5	1	1	1	7	7	7	2	2	2	2	2
		2	2	2	2	2	2	6	6	6	6	3	3	3	3	3	3	3	3	3
			3	3	3	1	1	1	2	2	2	2	2	6	6	6	1	1	1	6
Page Fault	✓	✓	✓	✓		✓	✓	✓	✓	✓		✓	✓	✓		✓	✓			✓

LRU; frame = 4																				
LRU	**1**	**2**	**3**	**4**	**2**	**1**	**5**	**6**	**2**	**1**	**2**	**3**	**7**	**6**	**3**	**2**	**1**	**2**	**3**	**6**
	1	1	1	1	1	1	1	1	1	1	1	1	1	6	6	6	6	6	6	6
		2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
			3	3	3	3	5	5	5	5	5	3	3	3	3	3	3	3	3	3
				4	4	4	4	6	6	6	6	6	7	7	7	7	1	1	1	1
Page Fault	✓	✓	✓	✓			✓	✓				✓	✓	✓			✓			

FIFO; frame = 4																				
FIFO	**1**	**2**	**3**	**4**	**2**	**1**	**5**	**6**	**2**	**1**	**2**	**3**	**7**	**6**	**3**	**2**	**1**	**2**	**3**	**6**
	1	1	1	1	1	1	5	5	5	5	5	3	3	3	3	3	1	1	1	1
		2	2	2	2	2	2	6	6	6	6	6	7	7	7	7	7	7	3	3
			3	3	3	3	3	3	2	2	2	2	2	6	6	6	6	6	6	6
				4	4	4	4	4	4	1	1	1	1	1	1	2	2	2	2	2
Page Fault	✓	✓	✓	✓			✓	✓	✓	✓		✓	✓	✓		✓	✓		✓	

OPT; frame = 4																				
OPT	**1**	**2**	**3**	**4**	**2**	**1**	**5**	**6**	**2**	**1**	**2**	**3**	**7**	**6**	**3**	**2**	**1**	**2**	**3**	**6**
	1	1	1	1	1	1	1	1	1	1	1	1	7	7	7	7	1	1	1	1
		2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
			3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
				4	4	4	5	6	6	6	6	6	6	6	6	6	6	6	6	6
Page Fault	✓	✓	✓	✓			✓	✓					✓				✓			

Task 3

Make a program (by using any language you know) that simulates

1. LRU replacement
2. FIFO replacement

algorithms. Attach a source code of your program to the answer sheet.

Solve the TASK 2 with the same data by your programs.

(put your source code and results of implementation here)

LRU

```
#include <stdio.h>

int flame[3];

int main()
{
  int n[20] = {1,2,3,4,2,1,5,6,2,1,2,3,7,6,3,2,1,2,3,6};
  int i,j,r,s,t,index,flag1 = 0,flag2 = 0,count = 0,fs[3];
  int fsize = 3;

  for(i = 0;i < 3;i ++)
    {
      flame[i] = -1;
    }

  for(j = 0;j < 20;j ++)
    {
      flag1 = 0;
      flag2 = 0;
      for(i = 0;i < 3;i ++)
          {
            if(flame[i] == n[j])
```

```
                {
              flag1 = 1;
              flag2 = 1;

              break;
                }
            }

     if(flag1 == 0)
       {
         for(i = 0;i < 3;i ++)
            {
              if(flame[i] == -1)
                {
                  flame[i] = n[j];
                  flag2 = 1;

                  break;
                }
            }
       }

     if(flag2 == 0)
       {
         for(i = 0;i < 3;i ++)
            {
              fs[i] = 0;
            }

         r = j;

         for(s = 1;s <= fsize - 1;s ++)
            {
              r --;

              for(i = 0;i < 3;i ++)
                {
                  if(flame[i] == n[r])
                        fs[i] = 1;
                }
            }
         for(i = 0;i < 3;i ++)
            {
              if(fs[i] == 0)
                {
                  index = i;
                }
            }
         flame[index] = n[j];
```

```
      count ++;
    }
  for(t = 0;t < 3;t ++)
    {
      printf(" %d",flame[t]);
    }
  printf("\n");
    }
}
```

```
./a.out

1 -1 -1
 1 2 -1
 1 2 3
 4 2 3
 4 2 3
 4 2 1
 5 2 1
 5 6 1
 5 6 2
 1 6 2
 1 6 2
 1 3 2
 7 3 2
 7 3 6
 7 3 6
 2 3 6
 2 3 1
 2 3 1
 2 3 1
 2 3 6
```

FIFO

```
#include <stdio.h>

int flame[3];

int main()
{
  int page[20] = {1,2,3,4,2,1,5,6,2,1,2,3,7,6,3,2,1,2,3,6};
  int i,j,t = 0,flag1 = 0,flag2 = 0;
  int fsize = 3;

  for(i = 0;i < 3;i ++)
    {
      flame[i] = -1;
    }
  for(j = 0;j < 20;j ++)
    {
      flag1 = 0;
      flag2 = 0;

      for(i = 0;i < 20;i ++)
            {
              if(flame[i] == page[j])
                {
              flag1 = 1;
              flag2 = 1;

              break;
                }
            }

  if(flag1 == 0)
    {
      for(i = 0;i < fsize;i ++)
            {
              if(flame[i] == -1)
                {
                  flame[i] = page[j];
                  flag2 = 1;

                  break;
                }
            }
    }

  if(flag2 == 0)
    {
      flame[t] = page[j];
      t ++;
      if(t >= fsize){
            t = 0;
      }
    }

  for(i = 0;i < 3;i ++)
    {
      printf(" %d",flame[i]);
    }
  printf("\n");
    }
}

./a.out
```

```
1 -1 -1
 1 2 -1
 1 2 3
 4 2 3
 4 2 3
 4 1 3
 4 1 5
 6 1 5
 6 2 5
 6 2 1
 6 2 1
 3 2 1
 3 7 1
 3 7 6
 3 7 6
 2 7 6
 2 1 6
 2 1 6
 2 1 3
 6 1 3
```