Here, i introduce two kinds of page replacement algorithms. First one is FIFO, this is the simplest page replacement algorithm. When we need to replace one page, FIFO always choose the oldest one, just like the queue, first in first out.

Following is the code:

```
import random
class my_queue():
    def __init__(self, max_length):
        self.que = []
        self.max_length = max_length

    def get(self):
        temp =self.que[0]
        del self.que[0]
        return temp

    def exist(self, number):
        return number in self.que

    def put(self, number):
        if len(self.que) < self.max_length:
            self.que.append(number)
            return False
        else:
            if self.exist(number):
                return True
            else:
                temp = self.get()
                self.que.append(number)
                return False

def FIFO(frame_length, reference_length, reference_string=None):
    myQ = my_queue(frame_length)
    if not reference_string:
        reference_string = [random.randint(0,9) for i in
range(reference_length)]
    else:
        reference_string = reference_string
    print(reference_string)
    wrong = 0
    for i in reference_string:
        if not myQ.put(i):
            print(myQ.que)
            wrong += 1
    print(wrong)
```

```
FIFO(3,10,[7,0,1,2,0,3,0,4,2,3,0,3,2,1,2,0,1,7,0,1])
```

We assume frame size is 3 and the reference string is 7,0,1,2,0,3,0,4,2,3,0,3,2,1,2,0,1,7,0,1, the ouput would be:

```
reference string:  [7, 0, 1, 2, 0, 3, 0, 4, 2, 3, 0, 3, 2, 1, 2, 0, 1, 7, 0,
1]
page falut:
[7]
[7, 0]
[7, 0, 1]
[0, 1, 2]
[1, 2, 3]
[2, 3, 0]
[3, 0, 4]
[0, 4, 2]
[4, 2, 3]
[2, 3, 0]
[3, 0, 1]
[0, 1, 2]
[1, 2, 7]
[2, 7, 0]
[7, 0, 1]
total faluts:  15
```

The second algorithm is LRU: least-recently-used algorithm. When we need to relpace one page, always choose the page that has not been used for the longest period of time.

Following is code:

```
import random
class my_stack():
    def __init__(self, max_length):
        self.max_length = max_length
        self.sta = []

    def get(self):
        temp =self.sta[-1]
        del self.sta[-1]
        return temp

    def exist(self, number):
        return number in self.sta

    def put(self,number):
        if len(self.sta) < self.max_length:
            if len(self.sta) == 0:
                self.sta.append(number)
                return False
            self.sta.append(self.sta[-1])
            for i in range(len(self.sta)-1, 0, -1):
                self.sta[i] = self.sta[i-1]
            self.sta[0] = number
            return False
        else:
            if self.exist(number):
                index = self.sta.index(number)
                for i in range(index,0,-1):
                    self.sta[i] = self.sta[i-1]
                self.sta[0] = number
                return True
            else:
                self.get()
                self.sta.append(self.sta[-1])
                for i in range(len(self.sta)-1, 0, -1):
                    self.sta[i] = self.sta[i-1]
                self.sta[0] = number
                return False

def LRU(frame_length, reference_length, reference_string=None):
    myS = my_stack(frame_length)
    if not reference_string:
        reference_string = [random.randint(0,9) for i in
range(reference_length)]
    else:
        reference_string = reference_string
    print(reference_string)
    wrong = 0
    for i in reference_string:
        if not myS.put(i):
            print(myS.sta)
            wrong += 1
    print(wrong)

LRU(3,10,[7,0,1,2,0,3,0,4,2,3,0,3,2,1,2,0,1,7,0,1])
```

We assume frame size is 3 and the reference string is 7,0,1,2,0,3,0,4,2,3,0,3,2,1,2,0,1,7,0,1, the ouput would be:

```
reference string:  [7, 0, 1, 2, 0, 3, 0, 4, 2, 3, 0, 3, 2, 1, 2, 0, 1, 7, 0,
1]
page fault:
[7]
[0, 7]
[1, 0, 7]
[2, 1, 0]
```

```
[3, 0, 2]
[4, 0, 3]
[2, 4, 0]
[3, 2, 4]
[0, 3, 2]
[1, 2, 3]
[0, 2, 1]
[7, 1, 0]
total faluts:  12
```