Page and Frame Replacement Algorithms Program

โดย

ชื่อ บางกอก วาณิชยานนท์

รหัส 630610746

Computer Engineering

Chiang Mai University

คำแนะนำ: พัฒนาโปรแกรมของนักศึกษาเองเพื่อประเมินอัลกอริทึม Page and Frame Replacement แบบต่างๆ ให้ระบุเกณฑ์ที่ต้องการใช้เปรียบเทียบประสิทธิภาพของแต่ละอัลกอริทึม

แสดงโค้ดการเขียนโปรแกรมทั้งหมด ตามด้วยคำอธิบายสั้นๆ ของอัลกอริทึมทั้งสาม

The code and description of the First-In-First-Out (FIFO) algorithm (20 points)

In short: page เข้ามาเก็บใน frames ก่อน = ถูก replace ก่อน

Note: I believe my code is self-descriptive enough that minimal lines of comments are needed.

```
import { SimulationRecord } from "../types";

const increaseIndex = (index: number, limit: number) => {
    return (index + 1) % limit;
};

export const fifo = (
    referenceString: number[],
    frameCount: number
): SimulationRecord[] => {
    console.log(`FIFO, Frame Count: ${frameCount}`);
    const frames: number[] = new Array<number>(frameCount);
    const outputs: SimulationRecord[] = [];

    let nextReplaceIndex: number = 0;
    let pageFaultCounter: number = 0;

    for (let i = 0; i < referenceString.length; ++i) {
        const pageNumber = referenceString[i];
        const pageFault = !frames.includes(pageNumber);

        if (pageFault) {
            pageFaultCounter++;
            if (frameCount > 0) {
                frames[nextReplaceIndex] = pageNumber;
                nextReplaceIndex = increaseIndex(nextReplaceIndex, frameCount);
            }
        }
```

```
        const output: SimulationRecord = {
            index: i,
            pageNumber,
            frames: [...frames],
            pageFault,
            pageFaultCounter,
        };

        outputs.push(output);
    }

    console.log(
        `Total Page Faults: ${outputs[outputs.length - 1].pageFaultCounter}\n`
    );

    return outputs;
};
```

The code and description of the Optimal algorithm (20 points)

In short: สำหรับแต่ละ page ใน frames, ไล่ดูอนาคต referenceString ว่า page ใดจะถูกเรียกทีหลังสุด, replace page นั้น

Note: I believe my code is self-descriptive enough that minimal lines of comments are needed.

```
import { SimulationRecord } from "../types";

const findFurtestPageIndex = (
    currentIndex: number,
    referenceString: number[],
    frames: number[]
) => {
    let maxLength = 0;
    let maxPage = frames[0];
    let maxIndex = 0;

    for (let i = 0; i < frames.length; ++i) {
        const indexOfPage = referenceString.indexOf(
            frames[i],
            currentIndex + 1
        );
        if (indexOfPage == -1) {
            maxLength = indexOfPage;
            maxPage = frames[i];
            maxIndex = i;
```

```
            break;
        } else if (indexOfPage > maxLength) {
            maxLength = indexOfPage;
            maxPage = frames[i];
            maxIndex = i;
        }
    }

    return maxIndex;
};

export const optimal = (
    referenceString: number[],
    frameCount: number
): SimulationRecord[] => {
    console.log(`Optimal, Frame Count: ${frameCount}`);
    const frames: number[] = new Array<number>(frameCount);
    const outputs: SimulationRecord[] = [];

    let pageFaultCounter: number = 0;

    for (let i = 0; i < referenceString.length; ++i) {
        const pageNumber = referenceString[i];
        const pageFault = !frames.includes(pageNumber);

        if (pageFault) {
            pageFaultCounter++;
            if (frameCount > 0) {
                // find the page number that will not be used for longest period
of time
                let replaceIndex = findFurtestPageIndex(
                    i,
                    referenceString,
                    frames
                );

                // replace frame with that page with new page
                frames[replaceIndex] = pageNumber;
            }
        }

        const output: SimulationRecord = {
            index: i,
            pageNumber,
            frames: [...frames],
```

```
            pageFault,
            pageFaultCounter,
        };

        outputs.push(output);
    }

    console.log(
        `Total Page Faults: ${outputs[outputs.length - 1].pageFaultCounter}\n`
    );

    return outputs;
};
```

The code and description of the Least Recently Used (LRU) algorithm (20 points)

In short: สำหรับแต่ละ page ใน frames, เก็บ age ของแต่ละ page ว่าอยู่ใน frame มานานเท่าไหร่, เมื่อ page ถูกเรียกใช้จาก frame ให้ reset age เป็น 0, เมื่อจะ replace ให้ replace page ที่ age มากที่สุด

Note: I believe my code is self-descriptive enough that minimal lines of comments are needed.

```
import { SimulationRecord } from "../types";

const initFrameAge = (frameCount: number) => {
    const frameAges: number[] = new Array<number>(frameCount);
    for (let i = 0; i < frameCount; ++i) frameAges[i] = 2000000;

    return frameAges;
};

const countUpFrameAge = (frames: number[], frameAges: number[]) => {
    for (let i = 0; i < frameAges.length; ++i) {
        if (frames[i] != null) {
            frameAges[i]++;
        }
    }
};

const findLeastUsedPageIndex = (frames: number[], frameAges: number[]) => {
    let maxIndex = 0;

    const maxAge = Math.max(...frameAges);

    maxIndex = frameAges.indexOf(maxAge);
```

```
    return maxIndex;
};

export const lru = (
    referenceString: number[],
    frameCount: number
): SimulationRecord[] => {
    console.log(`LRU, Frame Count: ${frameCount}`);
    const frames: number[] = new Array<number>(frameCount);
    const frameAges: number[] = initFrameAge(frameCount);

    const outputs: SimulationRecord[] = [];

    let pageFaultCounter: number = 0;

    for (let i = 0; i < referenceString.length; ++i) {
        const pageNumber = referenceString[i];
        const pageFault = !frames.includes(pageNumber);

        if (pageFault) {
            pageFaultCounter++;

            if (frameCount > 0) {
                // find least used frame
                let replaceIndex = findLeastUsedPageIndex(frames, frameAges);

                frameAges[replaceIndex] = 0;

                // replace frame with that page with new page
                frames[replaceIndex] = pageNumber;
            }
        } else {
            frameAges[frames.indexOf(pageNumber)] = 0;
        }

        countUpFrameAge(frames, frameAges);

        const output: SimulationRecord = {
            index: i,
            pageNumber,
            frames: [...frames],
            pageFault,
            pageFaultCounter,
        };
```

```
        outputs.push(output);
    }

    console.log(
        `Total Page Faults: ${outputs[outputs.length - 1].pageFaultCounter}\n`
    );

    return outputs;
};
```

Experiments

1. **ชุดข้อมูลแรก first reference string dataset (Original)** (5 points)

สิ่งที่แสดงด้านล่างนี้คือการสร้าง reference string ชุดแรกโดยการสุ่ม มีความยาวอย่างน้อย 30 pages

```
8, 0, 2, 12, 1, 6, 12, 6, 3, 0, 2, 9, 6, 12, 7, 2, 15, 10, 5, 3, 3, 3, 10, 11, 6,
6, 5, 15, 8, 5,
```

อธิบายสมมติฐานที่นักศึกษาใช้ในการสร้าง reference string เช่นความหลากหลาย การถูกอ้างซ้ำเป็นต้น (5 points)

Reference string ประกอบด้วย page number ที่อยู่ในช่วง [0,15] โดย page number ในช่วง [0,7] จะมีโอกาสถูกสุ่มออกมา 0.67 และ page number ในช่วง [8,15] จะมีโอกาสถูกสุ่มออกมา 0.33 นั่นคือความหนาแน่นของเลขในช่วง [0,7] จะมากกว่า สังเกตได้จากเลข 3 ที่อยู่ติดกัน 3 ตัวเป็นต้น

แสดงผลการทดลองที่ได้จากโปรแกรมที่พัฒนาขึ้นโดยใช้ First-In-First-Out (FIFO) algorithm กับชุดข้อมูลแรก (5 points)

Frame = 6

index	pageNumber	frames	pageFault	pageFaultCounter
0	8	8,,,,,	true	1
1	0	8,0,,,,	true	2
2	2	8,0,2,,,	true	3
3	12	8,0,2,12,,	true	4
4	1	8,0,2,12,1,	true	5
5	6	8,0,2,12,1,6	true	6
6	12	8,0,2,12,1,6	false	6
7	6	8,0,2,12,1,6	false	6
8	3	3,0,2,12,1,6	true	7
9	0	3,0,2,12,1,6	false	7
10	2	3,0,2,12,1,6	false	7
11	9	3,9,2,12,1,6	true	8
12	6	3,9,2,12,1,6	false	8
13	12	3,9,2,12,1,6	false	8
14	7	3,9,7,12,1,6	true	9
15	2	3,9,7,2,1,6	true	10
16	15	3,9,7,2,15,6	true	11
17	10	3,9,7,2,15,10	true	12
18	5	5,9,7,2,15,10	true	13
19	3	5,3,7,2,15,10	true	14
20	3	5,3,7,2,15,10	false	14
21	3	5,3,7,2,15,10	false	14
22	10	5,3,7,2,15,10	false	14
23	11	5,3,11,2,15,10	true	15
24	6	5,3,11,6,15,10	true	16
25	6	5,3,11,6,15,10	false	16
26	5	5,3,11,6,15,10	false	16
27	15	5,3,11,6,15,10	false	16
28	8	5,3,11,6,8,10	true	17
29	5	5,3,11,6,8,10	false	17

All FIFO results; X = Frame, Y = Numbers of page faults when the algorithm ends

```
[
                    { x: 0, y: 30 },
                    { x: 1, y: 27 },
                    { x: 2, y: 26 },
                    { x: 3, y: 23 },
                    { x: 4, y: 23 },
                    { x: 5, y: 21 },
                    { x: 6, y: 17 },
                    { x: 7, y: 15 },
                    { x: 8, y: 14 },
                    { x: 9, y: 14 },
                    { x: 10, y: 14 },
                    { x: 11, y: 14 },
                    { x: 12, y: 14 },
                ]
```

แสดงผลการทดลองที่ได้จากโปรแกรมที่พัฒนาขึ้นโดยใช้ Optimal algorithm กับชุดข้อมูลแรก (5 points)

Frame = 6

index	pageNumber	frames	pageFault	pageFaultCounter
0	8	8,,,,,	true	1
1	0	8,0,,,,	true	2
2	2	8,0,2,,,	true	3
3	12	8,0,2,12,,	true	4
4	1	8,0,2,12,1,	true	5
5	6	8,0,2,12,6,	true	6
6	12	8,0,2,12,6,	false	6
7	6	8,0,2,12,6,	false	6
8	3	8,0,2,12,6,3	true	7
9	0	8,0,2,12,6,3	false	7
10	2	8,0,2,12,6,3	false	7
11	9	8,9,2,12,6,3	true	8
12	6	8,9,2,12,6,3	false	8
13	12	8,9,2,12,6,3	false	8
14	7	8,7,2,12,6,3	true	9
15	2	8,7,2,12,6,3	false	9
16	15	8,15,2,12,6,3	true	10
17	10	8,15,10,12,6,3	true	11
18	5	8,15,10,5,6,3	true	12
19	3	8,15,10,5,6,3	false	12
20	3	8,15,10,5,6,3	false	12
21	3	8,15,10,5,6,3	false	12
22	10	8,15,10,5,6,3	false	12
23	11	8,15,11,5,6,3	true	13
24	6	8,15,11,5,6,3	false	13
25	6	8,15,11,5,6,3	false	13
26	5	8,15,11,5,6,3	false	13
27	15	8,15,11,5,6,3	false	13
28	8	8,15,11,5,6,3	false	13
29	5	8,15,11,5,6,3	false	13

All Optimal results; X = Frame, Y = Numbers of page faults when the algorithm ends

```
[
                    { x: 0, y: 30 },
                    { x: 1, y: 27 },
                    { x: 2, y: 22 },
                    { x: 3, y: 19 },
                    { x: 4, y: 16 },
                    { x: 5, y: 14 },
                    { x: 6, y: 13 },
                    { x: 7, y: 13 },
                    { x: 8, y: 13 },
                    { x: 9, y: 13 },
                    { x: 10, y: 13 },
                    { x: 11, y: 13 },
                    { x: 12, y: 13 },
                ]
```

แสดงผลการทดลองที่ได้จากโปรแกรมที่พัฒนาขึ้นโดยใช้ Least Recently Used (LRU) algorithm กับชุดข้อมูลแรก (5 points)

Frame = 6

index	pageNumber	frames	pageFault	pageFaultCounter
0	8	8,,,,,	true	1
1	0	8,0,,,,	true	2
2	2	8,0,2,,,	true	3
3	12	8,0,2,12,,	true	4
4	1	8,0,2,12,1,	true	5
5	6	8,0,2,12,1,6	true	6
6	12	8,0,2,12,1,6	false	6
7	6	8,0,2,12,1,6	false	6
8	3	3,0,2,12,1,6	true	7
9	0	3,0,2,12,1,6	false	7
10	2	3,0,2,12,1,6	false	7
11	9	3,0,2,12,9,6	true	8
12	6	3,0,2,12,9,6	false	8
13	12	3,0,2,12,9,6	false	8
14	7	7,0,2,12,9,6	true	9
15	2	7,0,2,12,9,6	false	9
16	15	7,15,2,12,9,6	true	10
17	10	7,15,2,12,10,6	true	11
18	5	7,15,2,12,10,5	true	12
19	3	7,15,2,3,10,5	true	13
20	3	7,15,2,3,10,5	false	13
21	3	7,15,2,3,10,5	false	13
22	10	7,15,2,3,10,5	false	13
23	11	11,15,2,3,10,5	true	14
24	6	11,15,6,3,10,5	true	15
25	6	11,15,6,3,10,5	false	15
26	5	11,15,6,3,10,5	false	15
27	15	11,15,6,3,10,5	false	15
28	8	11,15,6,8,10,5	true	16
29	5	11,15,6,8,10,5	false	16

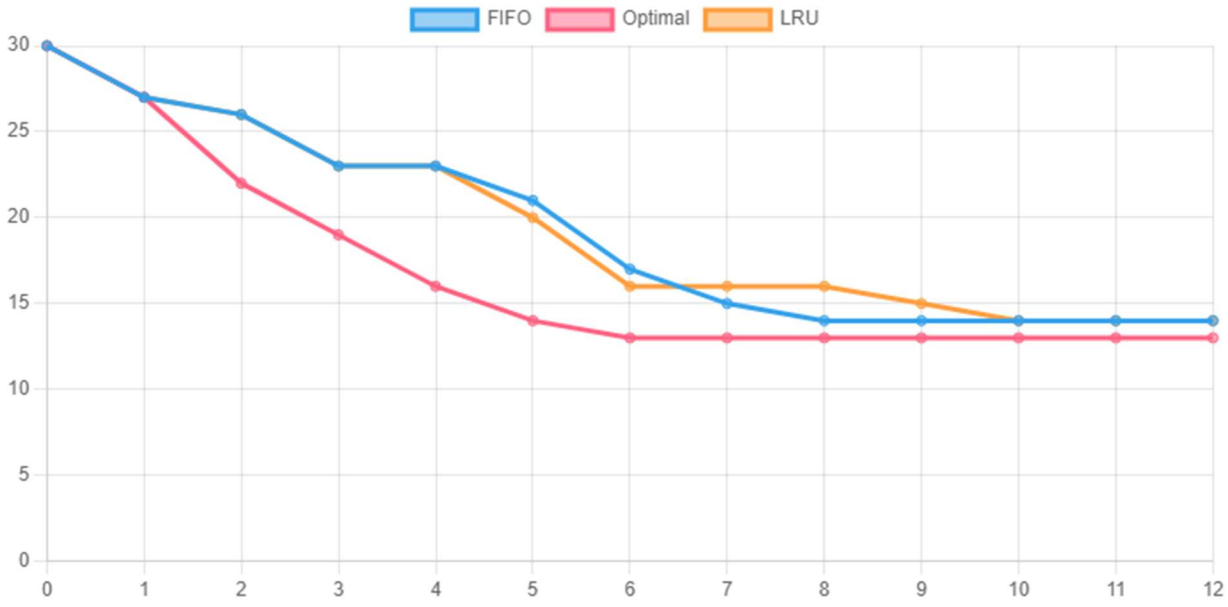
All LRU results; X = Frame, Y = Numbers of page faults when the algorithm ends

```
[
                    { x: 0, y: 30 },
                    { x: 1, y: 27 },
                    { x: 2, y: 26 },
                    { x: 3, y: 23 },
                    { x: 4, y: 23 },
                    { x: 5, y: 20 },
                    { x: 6, y: 16 },
                    { x: 7, y: 16 },
                    { x: 8, y: 16 },
                    { x: 9, y: 15 },
                    { x: 10, y: 14 },
                    { x: 11, y: 14 },
                    { x: 12, y: 14 },
                ]
```

ตารางสรุป ผลการทดลอง

frames	FIFO	Optimal	LRU
0	30	30	30
1	27	27	27
2	26	22	26
3	23	19	23
4	23	16	23
5	21	14	20
6	17	13	16
7	15	13	16
8	14	13	16
9	14	13	15
10	14	13	14
11	14	13	14
12	14	13	14

Graph สรุปผลการทดลอง



2. **ชุดข้อมูลแรก second reference string dataset (Original)** (5 points)

สิ่งที่แสดงด้านล่างนี้คือการสร้าง reference string ชุดแรกโดยการสุ่ม มีความยาวอย่างน้อย 50 pages

```
8, 6, 12, 2, 6, 14, 2, 14, 10, 4, 7, 5, 12, 2, 6, 12, 15, 7, 4, 6, 3, 10,
14, 1, 7, 11, 7, 0, 4, 7, 4, 9, 7, 6, 6, 11, 13, 12, 15, 1, 1, 0, 15, 4, 6,
0, 5, 2, 3, 4,
```

อธิบายสมมติฐานที่นักศึกษาใช้ในการสร้าง reference string เช่นความหลากหลาย การถูกอ้างซ้ำเป็นต้น (5 points)

Reference string ประกอบด้วย page number ที่อยู่ในช่วง [0,15] โดย page number ในช่วง [0,7] จะมีโอกาสถูกสุ่มออกมา 0.67 และ page number ในช่วง [8,15] จะมีโอกาสถูกสุ่มออกมา 0.33 นั่นคือความหนาแน่นของเลขในช่วง [0,7] จะมากกว่า สังเกตได้จากเลข 7 ที่โผล่มาเยอะมากในแถวที่ 2

แสดงผลการทดลองที่ได้จากโปรแกรมที่พัฒนาขึ้นโดยใช้ First-In-First-Out (FIFO) algorithm กับชุดข้อมูลที่ 2 (5 points)

index	pageNumber	frames	pageFault	pageFaultCounter
0	8	8,,,,,	true	1
1	6	8,6,,,,	true	2
2	12	8,6,12,,,	true	3
3	2	8,6,12,2,,	true	4
4	6	8,6,12,2,,	false	4
5	14	8,6,12,2,14,	true	5
6	2	8,6,12,2,14,	false	5
7	14	8,6,12,2,14,	false	5
8	10	8,6,12,2,14,10	true	6
9	4	4,6,12,2,14,10	true	7
10	7	4,7,12,2,14,10	true	8
11	5	4,7,5,2,14,10	true	9
12	12	4,7,5,12,14,10	true	10
13	2	4,7,5,12,2,10	true	11
14	6	4,7,5,12,2,6	true	12
15	12	4,7,5,12,2,6	false	12
16	15	15,7,5,12,2,6	true	13
17	7	15,7,5,12,2,6	false	13
18	4	15,4,5,12,2,6	true	14
19	6	15,4,5,12,2,6	false	14
20	3	15,4,3,12,2,6	true	15
21	10	15,4,3,10,2,6	true	16
22	14	15,4,3,10,14,6	true	17
23	1	15,4,3,10,14,1	true	18
24	7	7,4,3,10,14,1	true	19
25	11	7,11,3,10,14,1	true	20
26	7	7,11,3,10,14,1	false	20
27	0	7,11,0,10,14,1	true	21
28	4	7,11,0,4,14,1	true	22
29	7	7,11,0,4,14,1	false	22
30	4	7,11,0,4,14,1	false	22
31	9	7,11,0,4,9,1	true	23
32	7	7,11,0,4,9,1	false	23
33	6	7,11,0,4,9,6	true	24
34	6	7,11,0,4,9,6	false	24
35	11	7,11,0,4,9,6	false	24
36	13	13,11,0,4,9,6	true	25
37	12	13,12,0,4,9,6	true	26
38	15	13,12,15,4,9,6	true	27

39	1	13,12,15,1,9,6	true	28
40	1	13,12,15,1,9,6	false	28
41	0	13,12,15,1,0,6	true	29
42	15	13,12,15,1,0,6	false	29
43	4	13,12,15,1,0,4	true	30
44	6	6,12,15,1,0,4	true	31
45	0	6,12,15,1,0,4	false	31
46	5	6,5,15,1,0,4	true	32
47	2	6,5,2,1,0,4	true	33
48	3	6,5,2,3,0,4	true	34
49	4	6,5,2,3,0,4	false	34

All FIFO results; X = Frame, Y = Numbers of page faults when the algorithm ends

```
[
                    { x: 0, y: 50 },
                    { x: 1, y: 48 },
                    { x: 2, y: 44 },
                    { x: 3, y: 39 },
                    { x: 4, y: 38 },
                    { x: 5, y: 35 },
                    { x: 6, y: 34 },
                    { x: 7, y: 31 },
                    { x: 8, y: 28 },
                    { x: 9, y: 22 },
                    { x: 10, y: 22 },
                    { x: 11, y: 20 },
                    { x: 12, y: 20 },
                  ]
```

แสดงผลการทดลองที่ได้จากโปรแกรมที่พัฒนาขึ้นโดยใช้ Optimal algorithm กับชุดข้อมูลที่ 2 (5 points)

Frame = 6

index	pageNumber	frames	pageFault	pageFaultCounter
0	8	8,,,,,	true	1
1	6	6,,,,,	true	2
2	12	6,12,,,,	true	3
3	2	6,12,2,,,	true	4
4	6	6,12,2,,,	false	4
5	14	6,12,2,14,,	true	5
6	2	6,12,2,14,,	false	5
7	14	6,12,2,14,,	false	5
8	10	6,12,2,14,10,	true	6
9	4	6,12,2,14,10,4	true	7
10	7	6,12,2,7,10,4	true	8
11	5	6,12,2,7,5,4	true	9
12	12	6,12,2,7,5,4	false	9
13	2	6,12,2,7,5,4	false	9
14	6	6,12,2,7,5,4	false	9
15	12	6,12,2,7,5,4	false	9
16	15	6,12,15,7,5,4	true	10
17	7	6,12,15,7,5,4	false	10
18	4	6,12,15,7,5,4	false	10
19	6	6,12,15,7,5,4	false	10
20	3	6,12,15,7,3,4	true	11
21	10	6,12,15,7,10,4	true	12
22	14	6,12,15,7,14,4	true	13
23	1	6,12,15,7,1,4	true	14
24	7	6,12,15,7,1,4	false	14
25	11	6,12,15,7,11,4	true	15
26	7	6,12,15,7,11,4	false	15
27	0	6,12,0,7,11,4	true	16
28	4	6,12,0,7,11,4	false	16
29	7	6,12,0,7,11,4	false	16
30	4	6,12,0,7,11,4	false	16
31	9	6,12,0,7,11,9	true	17
32	7	6,12,0,7,11,9	false	17
33	6	6,12,0,7,11,9	false	17
34	6	6,12,0,7,11,9	false	17
35	11	6,12,0,7,11,9	false	17
36	13	6,12,0,13,11,9	true	18
37	12	6,12,0,13,11,9	false	18
38	15	6,15,0,13,11,9	true	19

39	1	6,15,0,1,11,9	true	20
40	1	6,15,0,1,11,9	false	20
41	0	6,15,0,1,11,9	false	20
42	15	6,15,0,1,11,9	false	20
43	4	6,4,0,1,11,9	true	21
44	6	6,4,0,1,11,9	false	21
45	0	6,4,0,1,11,9	false	21
46	5	5,4,0,1,11,9	true	22
47	2	2,4,0,1,11,9	true	23
48	3	3,4,0,1,11,9	true	24
49	4	3,4,0,1,11,9	false	24

All Optimal results; X = Frame, Y = Numbers of page faults when the algorithm ends

```
[
                    { x: 0, y: 50 },
                    { x: 1, y: 48 },
                    { x: 2, y: 36 },
                    { x: 3, y: 31 },
                    { x: 4, y: 28 },
                    { x: 5, y: 26 },
                    { x: 6, y: 24 },
                    { x: 7, y: 22 },
                    { x: 8, y: 20 },
                    { x: 9, y: 19 },
                    { x: 10, y: 18 },
                    { x: 11, y: 17 },
                    { x: 12, y: 16 },
                  ]
```

แสดงผลการทดลองที่ได้จากโปรแกรมที่พัฒนาขึ้นโดยใช้ Least Recently Used (LRU) algorithm กับชุดข้อมูลที่ 2 (5 points)

Frame = 6

index	pageNumber	frames	pageFault	pageFaultCounter
0	8	8,,,,,	true	1
1	6	8,6,,,,	true	2
2	12	8,6,12,,,	true	3
3	2	8,6,12,2,,	true	4
4	6	8,6,12,2,,	false	4
5	14	8,6,12,2,14,	true	5
6	2	8,6,12,2,14,	false	5
7	14	8,6,12,2,14,	false	5
8	10	8,6,12,2,14,10	true	6
9	4	4,6,12,2,14,10	true	7
10	7	4,6,7,2,14,10	true	8
11	5	4,5,7,2,14,10	true	9
12	12	4,5,7,12,14,10	true	10
13	2	4,5,7,12,2,10	true	11
14	6	4,5,7,12,2,6	true	12
15	12	4,5,7,12,2,6	false	12
16	15	15,5,7,12,2,6	true	13
17	7	15,5,7,12,2,6	false	13
18	4	15,4,7,12,2,6	true	14
19	6	15,4,7,12,2,6	false	14
20	3	15,4,7,12,3,6	true	15
21	10	15,4,7,10,3,6	true	16
22	14	14,4,7,10,3,6	true	17
23	1	14,4,1,10,3,6	true	18
24	7	14,7,1,10,3,6	true	19
25	11	14,7,1,10,3,11	true	20
26	7	14,7,1,10,3,11	false	20
27	0	14,7,1,10,0,11	true	21
28	4	14,7,1,4,0,11	true	22
29	7	14,7,1,4,0,11	false	22
30	4	14,7,1,4,0,11	false	22
31	9	9,7,1,4,0,11	true	23
32	7	9,7,1,4,0,11	false	23
33	6	9,7,6,4,0,11	true	24
34	6	9,7,6,4,0,11	false	24
35	11	9,7,6,4,0,11	false	24
36	13	9,7,6,4,13,11	true	25

37	12	9,7,6,12,13,11	true	26
38	15	15,7,6,12,13,11	true	27
39	1	15,1,6,12,13,11	true	28
40	1	15,1,6,12,13,11	false	28
41	0	15,1,0,12,13,11	true	29
42	15	15,1,0,12,13,11	false	29
43	4	15,1,0,12,13,4	true	30
44	6	15,1,0,12,6,4	true	31
45	0	15,1,0,12,6,4	false	31
46	5	15,1,0,5,6,4	true	32
47	2	15,2,0,5,6,4	true	33
48	3	3,2,0,5,6,4	true	34
49	4	3,2,0,5,6,4	false	34

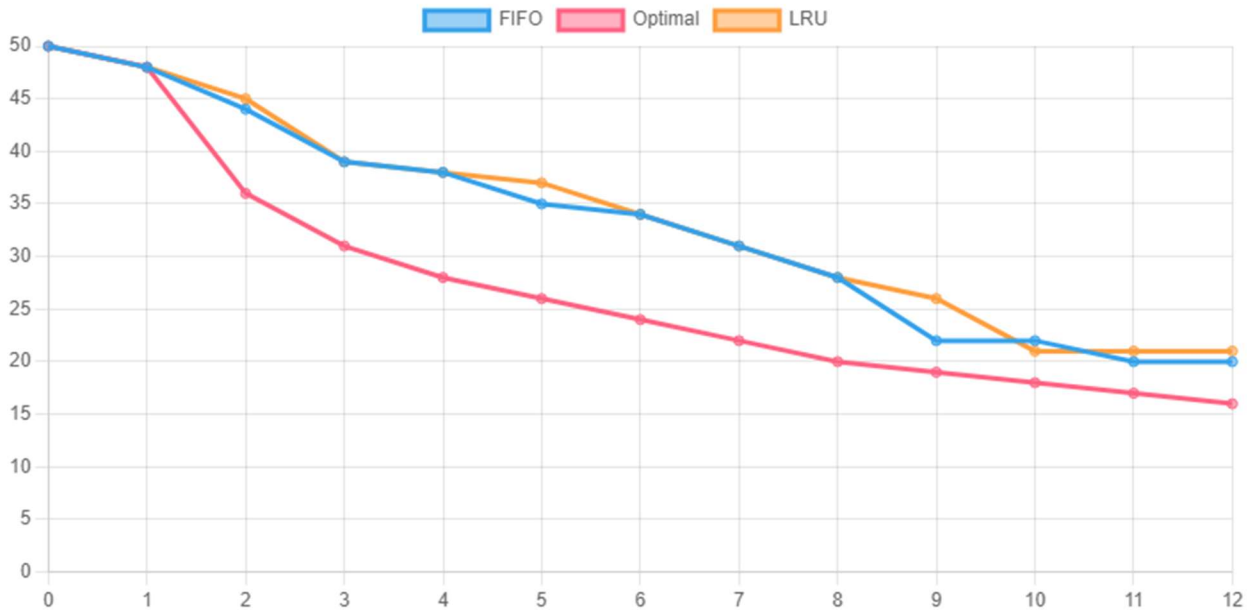
All LRU results; X = Frame, Y = Numbers of page faults when the algorithm ends

```
[
                    { x: 0, y: 50 },
                    { x: 1, y: 48 },
                    { x: 2, y: 45 },
                    { x: 3, y: 39 },
                    { x: 4, y: 38 },
                    { x: 5, y: 37 },
                    { x: 6, y: 34 },
                    { x: 7, y: 31 },
                    { x: 8, y: 28 },
                    { x: 9, y: 26 },
                    { x: 10, y: 21 },
                    { x: 11, y: 21 },
                    { x: 12, y: 21 },
                ]
```

ตารางสรุป ผลการทดลอง

frames	FIFO	Optimal	LRU
0	50	50	50
1	48	48	48
2	44	36	45
3	39	31	39
4	38	28	38
5	35	26	37
6	34	24	34
7	31	22	31
8	28	20	28
9	22	19	26
10	22	18	21
11	20	17	21
12	20	16	21

Graph สรุปผลการทดลอง



3. **ชุดข้อมูลแรก third reference string dataset (Original)** (5 points)

สิ่งที่แสดงด้านล่างนี้คือการสร้าง reference string ชุดแรกโดยการสุ่ม มีความยาวอย่างน้อย 100 pages

```
0, 0, 7, 12, 7, 4, 6, 4, 6, 0, 14, 2, 4, 8, 15, 0, 3, 15, 7, 2, 9, 14, 5, 3,
1, 6, 1, 1, 7, 3, 2, 3, 11, 11, 4, 6, 13, 12, 12, 3, 11, 8, 9, 3, 10, 13, 6,
2, 3, 6, 4, 5, 6, 15, 2, 5, 12, 15, 0, 4, 10, 1, 10, 4, 1, 1, 7, 7, 2, 0, 1,
14, 7, 4, 2, 1, 5, 2, 13, 0, 0, 1, 5, 9, 12, 5, 2, 6, 1, 6, 15, 2, 4, 4, 4,
1, 4, 0, 7, 3,
```

อธิบายสมมติฐานที่นักศึกษาใช้ในการสร้าง reference string เช่นความหลากหลาย การถูกอ้างซ้ำเป็นต้น (5 points)

Reference string ประกอบด้วย page number ที่อยู่ในช่วง [0,15] โดย page number ในช่วง [0,7] จะมีโอกาสถูกสุ่มออกมา 0.67 และ page number ในช่วง [8,15] จะมีโอกาสถูกสุ่มออกมา 0.33 นั่นคือความหนาแน่นของเลขในช่วง [0,7] จะมากกว่า

แสดงผลการทดลองที่ได้จากโปรแกรมที่พัฒนาขึ้นโดยใช้ First-In-First-Out (FIFO) algorithm กับชุดข้อมูลที่ 3 (5 points)

Frame = 6

index	pageNumber	frames	pageFault	pageFaultCounter
0	0	0,,,,,	true	1
1	0	0,,,,,	false	1
2	7	0,7,,,,	true	2
3	12	0,7,12,,,	true	3
4	7	0,7,12,,,	false	3
5	4	0,7,12,4,,	true	4
6	6	0,7,12,4,6,	true	5
7	4	0,7,12,4,6,	false	5
8	6	0,7,12,4,6,	false	5
9	0	0,7,12,4,6,	false	5
10	14	0,7,12,4,6,14	true	6
11	2	2,7,12,4,6,14	true	7
12	4	2,7,12,4,6,14	false	7
13	8	2,8,12,4,6,14	true	8
14	15	2,8,15,4,6,14	true	9
15	0	2,8,15,0,6,14	true	10
16	3	2,8,15,0,3,14	true	11
17	15	2,8,15,0,3,14	false	11
18	7	2,8,15,0,3,7	true	12
19	2	2,8,15,0,3,7	false	12
20	9	9,8,15,0,3,7	true	13
21	14	9,14,15,0,3,7	true	14
22	5	9,14,5,0,3,7	true	15
23	3	9,14,5,0,3,7	false	15
24	1	9,14,5,1,3,7	true	16
25	6	9,14,5,1,6,7	true	17
26	1	9,14,5,1,6,7	false	17
27	1	9,14,5,1,6,7	false	17
28	7	9,14,5,1,6,7	false	17
29	3	9,14,5,1,6,3	true	18
30	2	2,14,5,1,6,3	true	19
31	3	2,14,5,1,6,3	false	19
32	11	2,11,5,1,6,3	true	20
33	11	2,11,5,1,6,3	false	20
34	4	2,11,4,1,6,3	true	21
35	6	2,11,4,1,6,3	false	21
36	13	2,11,4,13,6,3	true	22

37	12	2,11,4,13,12,3	true	23
38	12	2,11,4,13,12,3	false	23
39	3	2,11,4,13,12,3	false	23
40	11	2,11,4,13,12,3	false	23
41	8	2,11,4,13,12,8	true	24
42	9	9,11,4,13,12,8	true	25
43	3	9,3,4,13,12,8	true	26
44	10	9,3,10,13,12,8	true	27
45	13	9,3,10,13,12,8	false	27
46	6	9,3,10,6,12,8	true	28
47	2	9,3,10,6,2,8	true	29
48	3	9,3,10,6,2,8	false	29
49	6	9,3,10,6,2,8	false	29
50	4	9,3,10,6,2,4	true	30
51	5	5,3,10,6,2,4	true	31
52	6	5,3,10,6,2,4	false	31
53	15	5,15,10,6,2,4	true	32
54	2	5,15,10,6,2,4	false	32
55	5	5,15,10,6,2,4	false	32
56	12	5,15,12,6,2,4	true	33
57	15	5,15,12,6,2,4	false	33
58	0	5,15,12,0,2,4	true	34
59	4	5,15,12,0,2,4	false	34
60	10	5,15,12,0,10,4	true	35
61	1	5,15,12,0,10,1	true	36
62	10	5,15,12,0,10,1	false	36
63	4	4,15,12,0,10,1	true	37
64	1	4,15,12,0,10,1	false	37
65	1	4,15,12,0,10,1	false	37
66	7	4,7,12,0,10,1	true	38
67	7	4,7,12,0,10,1	false	38
68	2	4,7,2,0,10,1	true	39
69	0	4,7,2,0,10,1	false	39
70	1	4,7,2,0,10,1	false	39
71	14	4,7,2,14,10,1	true	40
72	7	4,7,2,14,10,1	false	40
73	4	4,7,2,14,10,1	false	40
74	2	4,7,2,14,10,1	false	40
75	1	4,7,2,14,10,1	false	40
76	5	4,7,2,14,5,1	true	41
77	2	4,7,2,14,5,1	false	41
78	13	4,7,2,14,5,13	true	42
79	0	0,7,2,14,5,13	true	43
80	0	0,7,2,14,5,13	false	43
81	1	0,1,2,14,5,13	true	44

82	5	0,1,2,14,5,13	false	44
83	9	0,1,9,14,5,13	true	45
84	12	0,1,9,12,5,13	true	46
85	5	0,1,9,12,5,13	false	46
86	2	0,1,9,12,2,13	true	47
87	6	0,1,9,12,2,6	true	48
88	1	0,1,9,12,2,6	false	48
89	6	0,1,9,12,2,6	false	48
90	15	15,1,9,12,2,6	true	49
91	2	15,1,9,12,2,6	false	49
92	4	15,4,9,12,2,6	true	50
93	4	15,4,9,12,2,6	false	50
94	4	15,4,9,12,2,6	false	50
95	1	15,4,1,12,2,6	true	51
96	4	15,4,1,12,2,6	false	51
97	0	15,4,1,0,2,6	true	52
98	7	15,4,1,0,7,6	true	53
99	3	15,4,1,0,7,3	true	54

All FIFO results; X = Frame, Y = Numbers of page faults when the algorithm ends

```
[
                    { x: 0, y: 100 },
                    { x: 1, y: 91 },
                    { x: 2, y: 82 },
                    { x: 3, y: 77 },
                    { x: 4, y: 70 },
                    { x: 5, y: 58 },
                    { x: 6, y: 54 },
                    { x: 7, y: 53 },
                    { x: 8, y: 50 },
                    { x: 9, y: 43 },
                    { x: 10, y: 38 },
                    { x: 11, y: 36 },
                    { x: 12, y: 28 },
                ]
```

แสดงผลการทดลองที่ได้จากโปรแกรมที่พัฒนาขึ้นโดยใช้ Optimal algorithm กับชุดข้อมูลที่ 3 (5 points)

Frame = 6

index	pageNumber	frames	pageFault	pageFaultCounter
0	0	0,,,,,	true	1
1	0	0,,,,,	false	1
2	7	0,7,,,,	true	2
3	12	0,7,12,,,	true	3
4	7	0,7,12,,,	false	3
5	4	0,7,12,4,,	true	4
6	6	0,7,12,4,6,	true	5
7	4	0,7,12,4,6,	false	5
8	6	0,7,12,4,6,	false	5
9	0	0,7,12,4,6,	false	5
10	14	0,7,12,4,6,14	true	6
11	2	0,7,2,4,6,14	true	7
12	4	0,7,2,4,6,14	false	7
13	8	0,7,2,8,6,14	true	8
14	15	0,7,2,15,6,14	true	9
15	0	0,7,2,15,6,14	false	9
16	3	3,7,2,15,6,14	true	10
17	15	3,7,2,15,6,14	false	10
18	7	3,7,2,15,6,14	false	10
19	2	3,7,2,15,6,14	false	10
20	9	3,7,2,9,6,14	true	11
21	14	3,7,2,9,6,14	false	11
22	5	3,7,2,9,6,5	true	12
23	3	3,7,2,9,6,5	false	12
24	1	3,7,2,9,6,1	true	13
25	6	3,7,2,9,6,1	false	13
26	1	3,7,2,9,6,1	false	13
27	1	3,7,2,9,6,1	false	13
28	7	3,7,2,9,6,1	false	13
29	3	3,7,2,9,6,1	false	13
30	2	3,7,2,9,6,1	false	13
31	3	3,7,2,9,6,1	false	13
32	11	3,11,2,9,6,1	true	14
33	11	3,11,2,9,6,1	false	14
34	4	3,11,2,9,6,4	true	15
35	6	3,11,2,9,6,4	false	15
36	13	3,11,2,9,6,13	true	16
37	12	3,11,12,9,6,13	true	17
38	12	3,11,12,9,6,13	false	17

39	3	3,11,12,9,6,13	false	17
40	11	3,11,12,9,6,13	false	17
41	8	3,8,12,9,6,13	true	18
42	9	3,8,12,9,6,13	false	18
43	3	3,8,12,9,6,13	false	18
44	10	3,10,12,9,6,13	true	19
45	13	3,10,12,9,6,13	false	19
46	6	3,10,12,9,6,13	false	19
47	2	3,10,12,2,6,13	true	20
48	3	3,10,12,2,6,13	false	20
49	6	3,10,12,2,6,13	false	20
50	4	4,10,12,2,6,13	true	21
51	5	4,10,12,2,6,5	true	22
52	6	4,10,12,2,6,5	false	22
53	15	4,10,12,2,15,5	true	23
54	2	4,10,12,2,15,5	false	23
55	5	4,10,12,2,15,5	false	23
56	12	4,10,12,2,15,5	false	23
57	15	4,10,12,2,15,5	false	23
58	0	4,10,12,2,0,5	true	24
59	4	4,10,12,2,0,5	false	24
60	10	4,10,12,2,0,5	false	24
61	1	4,10,1,2,0,5	true	25
62	10	4,10,1,2,0,5	false	25
63	4	4,10,1,2,0,5	false	25
64	1	4,10,1,2,0,5	false	25
65	1	4,10,1,2,0,5	false	25
66	7	4,7,1,2,0,5	true	26
67	7	4,7,1,2,0,5	false	26
68	2	4,7,1,2,0,5	false	26
69	0	4,7,1,2,0,5	false	26
70	1	4,7,1,2,0,5	false	26
71	14	4,7,1,2,14,5	true	27
72	7	4,7,1,2,14,5	false	27
73	4	4,7,1,2,14,5	false	27
74	2	4,7,1,2,14,5	false	27
75	1	4,7,1,2,14,5	false	27
76	5	4,7,1,2,14,5	false	27
77	2	4,7,1,2,14,5	false	27
78	13	4,7,1,2,13,5	true	28
79	0	4,7,1,2,0,5	true	29
80	0	4,7,1,2,0,5	false	29
81	1	4,7,1,2,0,5	false	29
82	5	4,7,1,2,0,5	false	29
83	9	4,9,1,2,0,5	true	30

84	12	4,12,1,2,0,5	true	31
85	5	4,12,1,2,0,5	false	31
86	2	4,12,1,2,0,5	false	31
87	6	4,6,1,2,0,5	true	32
88	1	4,6,1,2,0,5	false	32
89	6	4,6,1,2,0,5	false	32
90	15	4,15,1,2,0,5	true	33
91	2	4,15,1,2,0,5	false	33
92	4	4,15,1,2,0,5	false	33
93	4	4,15,1,2,0,5	false	33
94	4	4,15,1,2,0,5	false	33
95	1	4,15,1,2,0,5	false	33
96	4	4,15,1,2,0,5	false	33
97	0	4,15,1,2,0,5	false	33
98	7	7,15,1,2,0,5	true	34
99	3	3,15,1,2,0,5	true	35

All Optimal results; X = Frame, Y = Numbers of page faults when the algorithm ends

```
[
                    { x: 0, y: 100 },
                    { x: 1, y: 91 },
                    { x: 2, y: 68 },
                    { x: 3, y: 54 },
                    { x: 4, y: 47 },
                    { x: 5, y: 40 },
                    { x: 6, y: 35 },
                    { x: 7, y: 31 },
                    { x: 8, y: 28 },
                    { x: 9, y: 26 },
                    { x: 10, y: 24 },
                    { x: 11, y: 22 },
                    { x: 12, y: 20 },
                ]
```

แสดงผลการทดลองที่ได้จากโปรแกรมที่พัฒนาขึ้นโดยใช้ Least Recently Used (LRU) algorithm กับชุดข้อมูลที่ 3 (5 points)

Frame = 6

index	pageNumber	frames	pageFault	pageFaultCounter
0	0	0,,,,,	true	1
1	0	0,,,,,	false	1
2	7	0,7,,,,	true	2
3	12	0,7,12,,,	true	3
4	7	0,7,12,,,	false	3
5	4	0,7,12,4,,	true	4
6	6	0,7,12,4,6,	true	5
7	4	0,7,12,4,6,	false	5
8	6	0,7,12,4,6,	false	5
9	0	0,7,12,4,6,	false	5
10	14	0,7,12,4,6,14	true	6
11	2	0,7,2,4,6,14	true	7
12	4	0,7,2,4,6,14	false	7
13	8	0,8,2,4,6,14	true	8
14	15	0,8,2,4,15,14	true	9
15	0	0,8,2,4,15,14	false	9
16	3	0,8,2,4,15,3	true	10
17	15	0,8,2,4,15,3	false	10
18	7	0,8,7,4,15,3	true	11
19	2	0,8,7,2,15,3	true	12
20	9	0,9,7,2,15,3	true	13
21	14	14,9,7,2,15,3	true	14
22	5	14,9,7,2,15,5	true	15
23	3	14,9,7,2,3,5	true	16
24	1	14,9,1,2,3,5	true	17
25	6	14,9,1,6,3,5	true	18
26	1	14,9,1,6,3,5	false	18
27	1	14,9,1,6,3,5	false	18
28	7	14,7,1,6,3,5	true	19
29	3	14,7,1,6,3,5	false	19
30	2	2,7,1,6,3,5	true	20
31	3	2,7,1,6,3,5	false	20
32	11	2,7,1,6,3,11	true	21
33	11	2,7,1,6,3,11	false	21
34	4	2,7,1,4,3,11	true	22
35	6	2,7,6,4,3,11	true	23
36	13	2,13,6,4,3,11	true	24

37	12	12,13,6,4,3,11	true	25
38	12	12,13,6,4,3,11	false	25
39	3	12,13,6,4,3,11	false	25
40	11	12,13,6,4,3,11	false	25
41	8	12,13,6,8,3,11	true	26
42	9	12,13,9,8,3,11	true	27
43	3	12,13,9,8,3,11	false	27
44	10	12,10,9,8,3,11	true	28
45	13	13,10,9,8,3,11	true	29
46	6	13,10,9,8,3,6	true	30
47	2	13,10,9,2,3,6	true	31
48	3	13,10,9,2,3,6	false	31
49	6	13,10,9,2,3,6	false	31
50	4	13,10,4,2,3,6	true	32
51	5	13,5,4,2,3,6	true	33
52	6	13,5,4,2,3,6	false	33
53	15	15,5,4,2,3,6	true	34
54	2	15,5,4,2,3,6	false	34
55	5	15,5,4,2,3,6	false	34
56	12	15,5,4,2,12,6	true	35
57	15	15,5,4,2,12,6	false	35
58	0	15,5,0,2,12,6	true	36
59	4	15,5,0,2,12,4	true	37
60	10	15,5,0,10,12,4	true	38
61	1	15,1,0,10,12,4	true	39
62	10	15,1,0,10,12,4	false	39
63	4	15,1,0,10,12,4	false	39
64	1	15,1,0,10,12,4	false	39
65	1	15,1,0,10,12,4	false	39
66	7	15,1,0,10,7,4	true	40
67	7	15,1,0,10,7,4	false	40
68	2	2,1,0,10,7,4	true	41
69	0	2,1,0,10,7,4	false	41
70	1	2,1,0,10,7,4	false	41
71	14	2,1,0,14,7,4	true	42
72	7	2,1,0,14,7,4	false	42
73	4	2,1,0,14,7,4	false	42
74	2	2,1,0,14,7,4	false	42
75	1	2,1,0,14,7,4	false	42
76	5	2,1,5,14,7,4	true	43
77	2	2,1,5,14,7,4	false	43
78	13	2,1,5,13,7,4	true	44
79	0	2,1,5,13,0,4	true	45
80	0	2,1,5,13,0,4	false	45
81	1	2,1,5,13,0,4	false	45

82	5	2,1,5,13,0,4	false	45
83	9	2,1,5,13,0,9	true	46
84	12	12,1,5,13,0,9	true	47
85	5	12,1,5,13,0,9	false	47
86	2	12,1,5,2,0,9	true	48
87	6	12,1,5,2,6,9	true	49
88	1	12,1,5,2,6,9	false	49
89	6	12,1,5,2,6,9	false	49
90	15	12,1,5,2,6,15	true	50
91	2	12,1,5,2,6,15	false	50
92	4	4,1,5,2,6,15	true	51
93	4	4,1,5,2,6,15	false	51
94	4	4,1,5,2,6,15	false	51
95	1	4,1,5,2,6,15	false	51
96	4	4,1,5,2,6,15	false	51
97	0	4,1,0,2,6,15	true	52
98	7	4,1,0,2,7,15	true	53
99	3	4,1,0,2,7,3	true	54

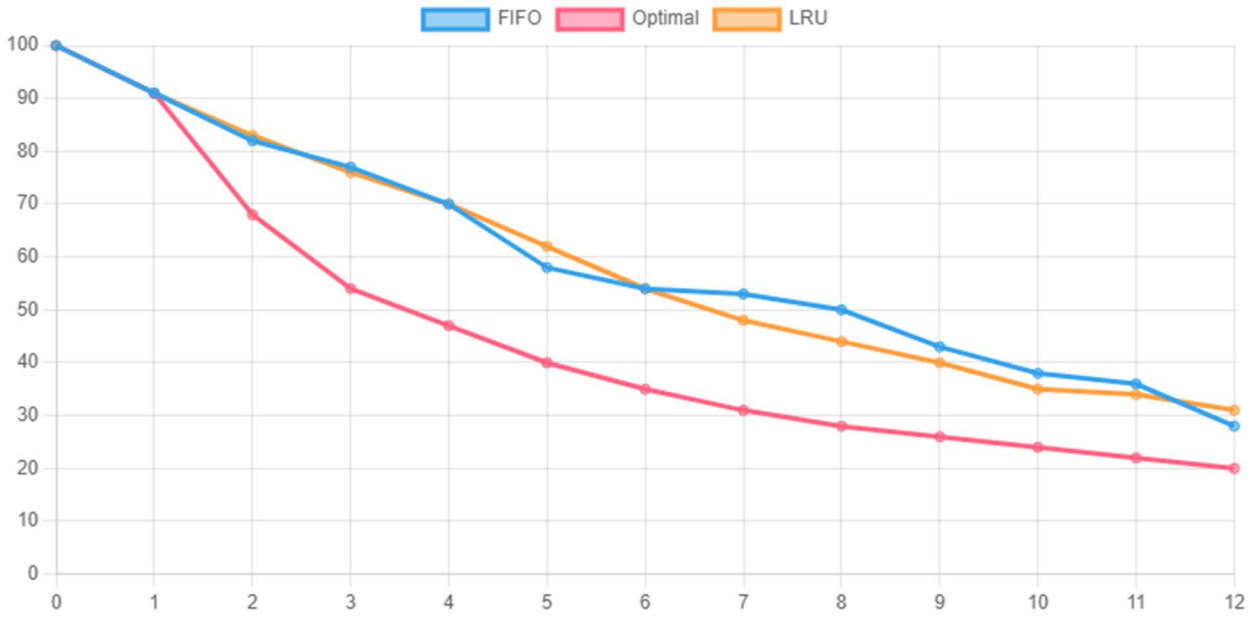
All LRU results; X = Frame, Y = Numbers of page faults when the algorithm ends

```
[
                    { x: 0, y: 100 },
                    { x: 1, y: 91 },
                    { x: 2, y: 83 },
                    { x: 3, y: 76 },
                    { x: 4, y: 70 },
                    { x: 5, y: 62 },
                    { x: 6, y: 54 },
                    { x: 7, y: 48 },
                    { x: 8, y: 44 },
                    { x: 9, y: 40 },
                    { x: 10, y: 35 },
                    { x: 11, y: 34 },
                    { x: 12, y: 31 },
                ]
```

ตารางสรุป ผลการทดลอง

frames	FIFO	Optimal	LRU
0	100	100	100
1	91	91	91
2	82	68	83
3	77	54	76
4	70	47	70
5	58	40	62
6	54	35	54
7	53	31	48
8	50	28	44
9	43	26	40
10	38	24	35
11	36	22	34
12	28	20	31

Graph สรุปผลการทดลอง



สรุปผลการทดลอง (20 points)

อธิบายการวิเคราะห์ ประสิทธิภาพ การทำงาน ของ Algorithm ทั้ง 3 โดยเปรียบเทียบ ผลการทดลองทั้งหมด ที่ได้จากการเขียนโปรแกรม

จากผลการทดลอง สังเกตได้ว่า Algorithm Optimal เจอ page fault น้อยที่สุด ส่วน FIFO และ LRU เจอกับ page fault ในปริมาณที่ใกล้เคียงกัน แต่โดยรวม LRU เจอน้อยกว่าอยู่เล็กน้อย
สรุปได้ว่าทั้ง FIFO และ LRU จะมีประสิทธิภาพดีในกรณีที่แตกต่างกันไป

จะเห็นว่าผลที่ได้แตกต่างจากใน slides ที่บอกว่า LRU มีประสิทธิภาพอยู่ระหว่าง Optimal และ FIFO ซึ่งผลอาจเป็นเช่นนี้ได้เพราะการสุ่ม reference string ที่ใช้