- The tables prepared in Task 2

Simpleloop

size = 50

	Hit rate	Hit count	Miss count	Total refereces	Clean evictions	Dirty evictions
FIFO	73.9628	8343	2937	11280	311	2576
LRU	75.3812	8503	2777	11280	217	2510
CLOCK	75.3635	8501	2779	11280	217	2512
ARC	75.5408	8521	2759	11280	195	2514
RAND	73.5638	8298	2982	11280	363	2569

size = 100

	Hit rate	Hit count	Miss count	Total refereces	Clean evictions	Dirty evictions
FIFO	75.5142	8518	2762	11280	179	2483
LRU	76.1702	8592	2688	11280	133	2455
CLOCK	76.0993	8584	2696	11280	140	2456
ARC	76.2411	8600	2680	11280	80	2500
RAND	75.4610	8512	2768	11280	193	2475

size = 150

	Hit rate	Hit count	Miss count	Total refereces	Clean evictions	Dirty evictions
FIFO	75.8954	8561	2719	11280	148	2421
LRU	76.2057	8596	2684	11280	130	2404
CLOCK	76.1968	8595	2685	11280	131	2404
ARC	76.4273	8621	2659	11280	25	2484
RAND	75.8865	8560	2720	11280	146	2424

size = 200

	Hit rate	Hit count	Miss count	Total refereces	Clean evictions	Dirty evictions
FIFO	75.9752	8570	2710	11280	143	2367
LRU	76.2057	8596	2684	11280	130	2354
CLOCK	76.1968	8595	2685	11280	130	2355

ARC	76.4273	8621	2659	11280	25	2434
RAND	75.9309	8565	2715	11280	143	2372

Blocked

size = 50

	Hit rate	Hit count	Miss count	Total refereces	Clean evictions	Dirty evictions
FIFO	95.7616	8450	374	8824	274	50
LRU	96.8155	8543	281	8824	210	21
CLOCK	96.7588	8538	286	8824	212	24
ARC	96.9288	8553	271	8824	197	24
RAND	95.4556	8423	401	8824	299	52

size = 100

	Hit rate	Hit count	Miss count	Total refereces	Clean evictions	Dirty evictions
FIFO	97.8241	8632	192	8824	75	17
LRU	98.0848	8655	169	8824	63	6
CLOCK	97.9828	8646	178	8824	71	7
ARC	97.8921	8638	186	8824	76	10
RAND	97.7221	8623	201	8824	87	14

size = 150

	Hit rate	Hit count	Miss count	Total refereces	Clean evictions	Dirty evictions
FIFO	98.1301	8659	165	8824	11	4
LRU	98.1981	8665	159	8824	8	1
CLOCK	98.1301	8659	165	8824	11	4
ARC	98.2094	8666	158	8824	8	0
RAND	98.1868	8664	160	8824	9	1

size = 200

	Hit rate	Hit count	Miss count	Total refereces	Clean evictions	Dirty evictions
FIFO	98.2094	8666	158	8824	0	0
LRU	98.2094	8666	158	8824	0	0

CLOCK	98.2094	8666	158	8824	0	0
ARC	98.2094	8666	158	8824	0	0
RAND	98.2094	8666	158	8824	0	0

Matmul

size = 50

	Hit rate	Hit count	Miss count	Total refereces	Clean evictions	Dirty evictions
FIFO	95.7547	8526	378	8904	274	54
LRU	96.6757	8608	296	8904	223	23
CLOCK	96.7093	8611	293	8904	219	24
ARC	96.8329	8622	282	8904	209	23
RAND	95.0809	8466	438	438	336	52

size = 100

	Hit rate	Hit count	Miss count	Total refereces	Clean evictions	Dirty evictions
FIFO	97.7089	8700	204	8904	86	18
LRU	97.9447	8721	183	8904	76	7
CLOCK	97.8774	8715	189	8904	82	7
ARC	97.7763	8706	198	8904	90	8
RAND	97.4843	8680	224	8904	106	18

size = 150

	Hit rate	Hit count	Miss count	Total refereces	Clean evictions	Dirty evictions
FIFO	97.9672	8723	181	8904	24	7
LRU	98.0795	8733	171	8904	20	1
CLOCK	98.0121	8727	177	8904	23	4
ARC	98.0795	8733	171	8904	20	1
RAND	97.9335	8720	184	8904	26	8

size = 200

	Hit rate	Hit count	Miss count	Total refereces	Clean evictions	Dirty evictions
FIFO	98.0795	8733	171	8904	0	0

LRU	98.0795	8733	171	8904	0	0
CLOCK	98.0795	8733	177	8904	0	0
ARC	98.0795	8733	171	8904	0	0
RAND	98.0795	8733	171	8904	0	0

fourth program

size = 50

	Hit rate	Hit count	Miss count	Total refereces	Clean evictions	Dirty evictions
FIFO	98.0795	6868	308	7176	217	41
LRU	96.6276	6934	242	7176	175	17
CLOCK	96.6276	6934	242	7176	171	21
ARC	96.7809	6945	231	7176	157	24
RAND	95.2620	6836	340	7176	244	46

size = 100

	Hit rate	Hit count	Miss count	Total refereces	Clean evictions	Dirty evictions
FIFO	97.5474	7000	176	7176	59	17
LRU	97.9097	7026	242	7176	45	5
CLOCK	97.7703	7016	160	7176	53	7
ARC	97.7982	7018	158	7176	46	12
RAND	97.5056	6997	179	7176	67	12

size = 150

	Hit rate	Hit count	Miss count	Total refereces	Clean evictions	Dirty evictions
FIFO	97.9794	7031	145	7176	0	0
LRU	97.9794	7031	145	7176	0	0
CLOCK	97.9794	7031	145	7176	0	0
ARC	97.9794	7031	145	7176	0	0
RAND	97.9794	7031	145	7176	0	0

size = 200

	Hit rate	Hit count	Miss count	Total refereces	Clean evictions	Dirty evictions
FIFO	97.9794	7031	145	7176	0	0
LRU	97.9794	7031	145	7176	0	0
CLOCK	97.9794	7031	145	7176	0	0
ARC	97.9794	7031	145	7176	0	0
RAND	97.9794	7031	145	7176	0	0

- Describe the fourth program of your choice and explain what you found interesting about its memory reference behaviour.

```c
#include <stdio.h>
#include <unistd.h>
#include <stdlib.h>

#define RECORD_SIZE 4096

struct krec {
    double d;
};

int main(int argc, char ** argv) {
    int i;
    struct krec *ptr = malloc(RECORD_SIZE *sizeof(struct krec));
    for(i = 0; i < RECORD_SIZE; i++) {
        ptr[i].d = (double)i;
    }
    free(ptr);
}

```

The program I wrote is a simple function which assign value to the attribute of each struct. The range of page numbers are small and

therefore the page are highly likely to be hit which will result in a a high hit rate for all the algorithms we have.

- One paragraph comparing the various algorithms in terms of the results you see in the tables. Do you notice any trends? Which ones are doing better in each case? Think about why and discuss.

lru and arc always doing better in following cases.

For lru, it will promote the most recent used pages to to head which has a higher possibility to be used again. Therefore, the pages are ordered from the most recent use pages to the least recent used pages. Since in this algorithm, we always evicted from the tail of the queue Therefore, the most recent used pages won't be delete until the least recent used pages have been deleted first. This avoid Belady and since the most recent used pages have a higher chance to be used again, it performs well in the rate of hitting pages.
For Arc, it is similar to lru, in which combines lru and lfu. It does not only promote the prior of the page that is most recently used but also the page that is used most frequently. Since both two kinds of pages are likely to be used again. Therefore, it also increases the rate of hitting pages.
For fifo, it suffers from Belady which lru and arc has avoided it, the recent used pages will be evicted first which might have a highly chance to be used again. Therefore, it declines the hit rates and this algorithm cannot perform better than either lru or arc.
For clock, it is similar to lru, it also avoids Belady and most of the time it performs better than fifo. The pages that has not be used for a long time will be evicted. However, for pages in the memory that have been accessed, the clock and lru algorithm both perform well, for pages that have accessed, it cannot accurately record their location like lru. Therefore, most of the time clock performs less better than lru.

- A second paragraph explaining the data you obtained for LRU and ARC as the size of memory increases. Specifically, comment on what you notice about the hit rate (does it increase or decrease?) when using different memory sizes. Do you notice any anomalies?

As size increase, both arc and lru has an increasing hit rate and decreasing dirty evictions. It is because both lru and arc has a similar data structure like queue. They promote the priority of the most recent used page and they are able to still in the slots for a long time which avoids Belady. The increasing in the memory size means

that more recently the used page will be stored in and it will increasing in the hit rate.
However, when size reaches 150, both hit rates of arc and lru stop increasing and will remain the same. It is because that we have enough space to place the pages that has been used in the memory and this result in the highest hitting rate.