Data Structures and Algorithms

Report on

Buddy Weighted Memory Allocator

And

One Bin Memory Allocator

Submitted By
Rahul Madhavan
Student, MTech AI

1. Description: Weighted Memory Allocation

We have created a buddy weighted memory allocator as well as a one bin allocator. Here are the header files to give some description of the structures and the functions.

```
#ifndef WEIGHTEDBUDDYMEMORYALLOCATOR_SPHEAP_H
#define WEIGHTEDBUDDYMEMORYALLOCATOR_SPHEAP_H

//Structure for getting stats
typedef struct heap_stats {
    int num_allocs;
    int num_deallocs;
    int splits;
    int recombines;
    int total_size_allocated;
    int total_size_requested;
    int total_size_of_heap;
} HeapStats;

//Below is the structure for the individual memory blocks
typedef struct memory_block {
    int kval; //stores the n in 2^n
    int tag;  //1 bit to indicate available (0) or reserved (1)
    int type; //Indicates whether the type is
    void *mem_address; //base address of the block. This is what will be returned
    struct memory_block *next;
    struct memory_block *prev;
    int memRequest;
} memBlock;

//Below are the memory bucket array structures which store the memory blocks as a linked list
typedef struct memory_bucket {
    int bucketSizeinB;
    int numMemBlocks;
    memBlock *head;
    memBlock *tail;
} memBucket;

//Finally, here is the spHeap structure which is actually called when initializing.
typedef struct spHeap {
    int num_buckets;
    int smallestBucketSize;
    int largestBucketSize;
    memBucket *memBuckets;
    HeapStats *stats;
    void* baseAddress;
} spHeap;

//We create a structure to store both the bucket number and the pointer at same time
typedef struct bucket_block {
    int bucket_num;
    memBlock *block;
} BucketBlock;

//Function which initializes memory
spHeap *initializeMemory(int heapBytes);
//Prints the current contents of the heap
void printHeap(spHeap *inputHeap);
//Below function is asked for allocation of memory
BucketBlock *allocateMemory(spHeap *inputHeap, int spaceRequired, int showErrors);
//Below function is used to free an individual memory element that had been earlier allocated
void freeMemory(spHeap *inputHeap, BucketBlock *bucketFreed);
//Below function is used to free the entire heap
void freeHeap(spHeap* inputHeap);

#endif //WEIGHTEDBUDDYMEMORYALLOCATOR_SPHEAP_H
```

Here are the definitions which are used to for SPHeap

```
#define SIZEOFHEAP 8 //(in 2^n MB)
#define MIN_ALLOCATABLE_BYTES 4 //(SIZEOF(INT) = 4
#define MAX_HEAP_SIZE 536870912
#define AVAILABLE 0
#define RESERVED 1
#define COMBINE31 0
#define COMBINE22 1
#define BUDDYLO 0
#define BUDDYHI 1
```

Here is the header used for the SPHeap tester

```

#ifndef WEIGHTEDBUDDYMEMORYALLOCATOR_SPHEAPTESTER_H
#define WEIGHTEDBUDDYMEMORYALLOCATOR_SPHEAPTESTER_H

#include "spHeap.h"
#include "helpers.h"
//Function to get a corrected possible spHeap size for any given size
int correctedSize(int memSizeinBytes);
//Function to get the bucket number in spHeap given the memory size
int bucket_num(int memSizeRequired);
//Function to get the bucket size given the bucket number
int get_bucket_size(int bucket_num);
//Function to check whether space is available in the input heap
BucketBlock *checkSpaceAvailableBucket(spHeap *inputHeap, int spaceRequired);
//Function to print a given memory block
void printMemBlock(memBlock *inputBlock);

#endif //WEIGHTEDBUDDYMEMORYALLOCATOR_SPHEAPTESTER_H
```

Here is the header used for the helpers used

```
#ifndef WEIGHTEDBUDDYMEMORYALLOCATOR_HELPERS_H
#define WEIGHTEDBUDDYMEMORYALLOCATOR_HELPERS_H
//Helper function to get the next multiple of eight, for ensuring alignment of memory allocated
int next_multiple_of8(int n);
//Function to get next highest power of two to allocate memory sizes in powers of 2
unsigned int nextPowerOf2(unsigned int n);
//Function to get 2^n
int two_power(int n);
//Function to print a bin
void printBin(int memSize);
//Function to check if a given number is a power of two
int isPowerOfTwo(int n);

#endif //WEIGHTEDBUDDYMEMORYALLOCATOR_HELPERS_H
```

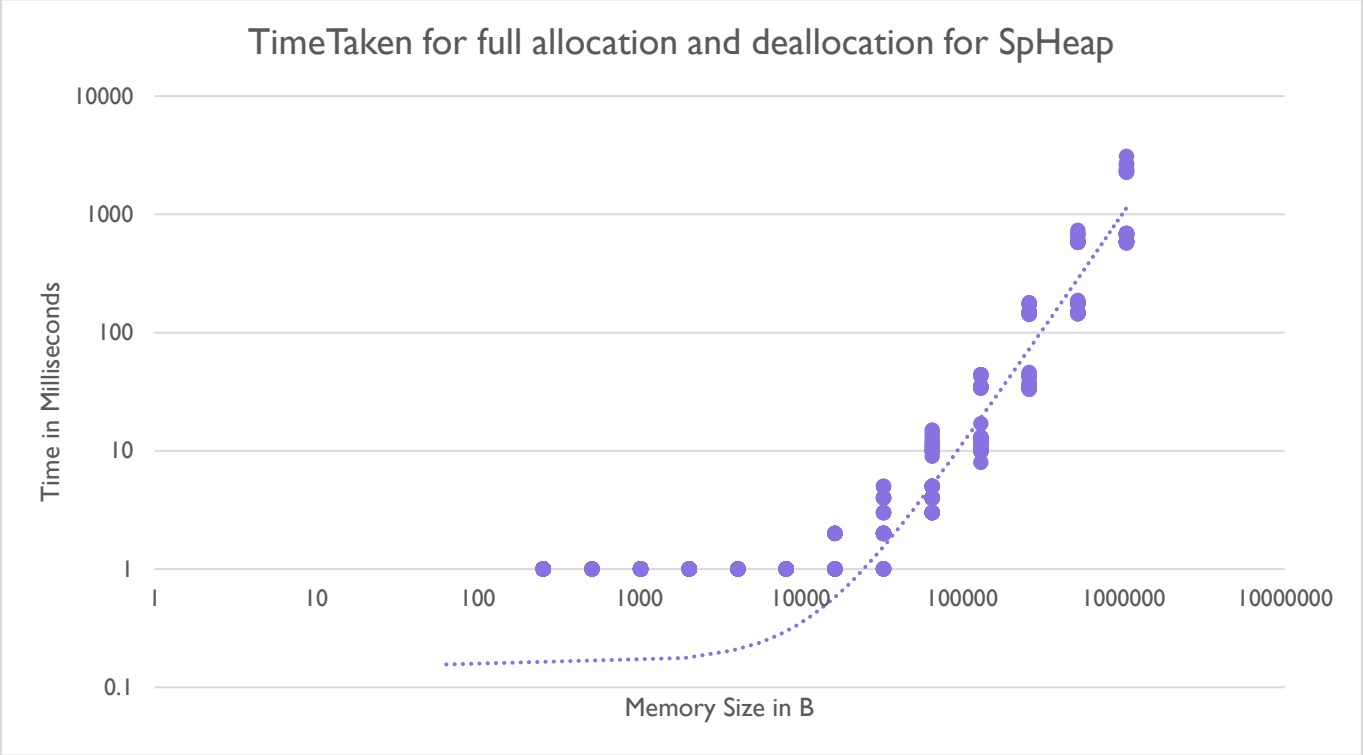
Now we will look at what are the oneBin functions

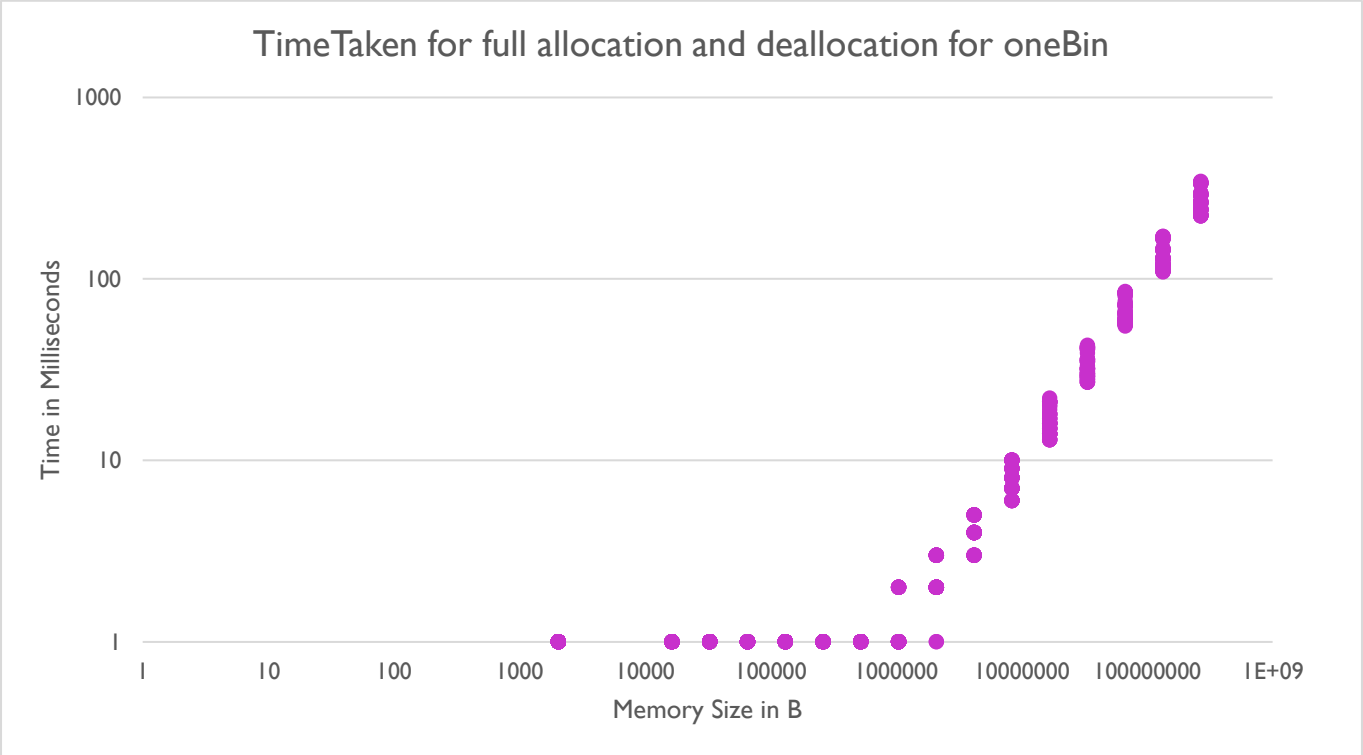
```
#ifndef WEIGHTEDBUDDYMEMORYALLOCATOR_ONEBIN_H
#define WEIGHTEDBUDDYMEMORYALLOCATOR_ONEBIN_H
//Defining the minimum bin size as below as we want space for 2 pointers and an int on 64 bit
machines
#define MINBIN_SIZE 24
//Structure to store the oneBin heap
typedef struct ob_want_heap{
    int total_size;
    int total_allocated;
    int onebin_size;
    int num_chunks;
    void* base_address;
    void* firstFree;
}oneBin;
//Structure to store each memory chunk. These are stored within the memory itself rather than
separately
typedef struct memChunk{
    int size;
    void* nextChunk;
    void* prevChunk;
} memChunk;
//Function to obtain memory of given size from heap
void* ob_wan_memory(oneBin* ob_heap);

//Function to initialize oneBin heap
oneBin* ob_start_kenobi(int memSize,int oneBinSize);

//Function to free a particular piece of memory when it's no longer in use.
void ob_free_la_mem(oneBin* ob, void* obis_memory);

//Function to shine light (print) on memory to see it's addresses and bins
void lightSaber(oneBin* ob);

//Function to free the entire heap;
void freeB(oneBin*ob);

#endif //WEIGHTEDBUDDYMEMORYALLOCATOR_ONEBIN_H
```

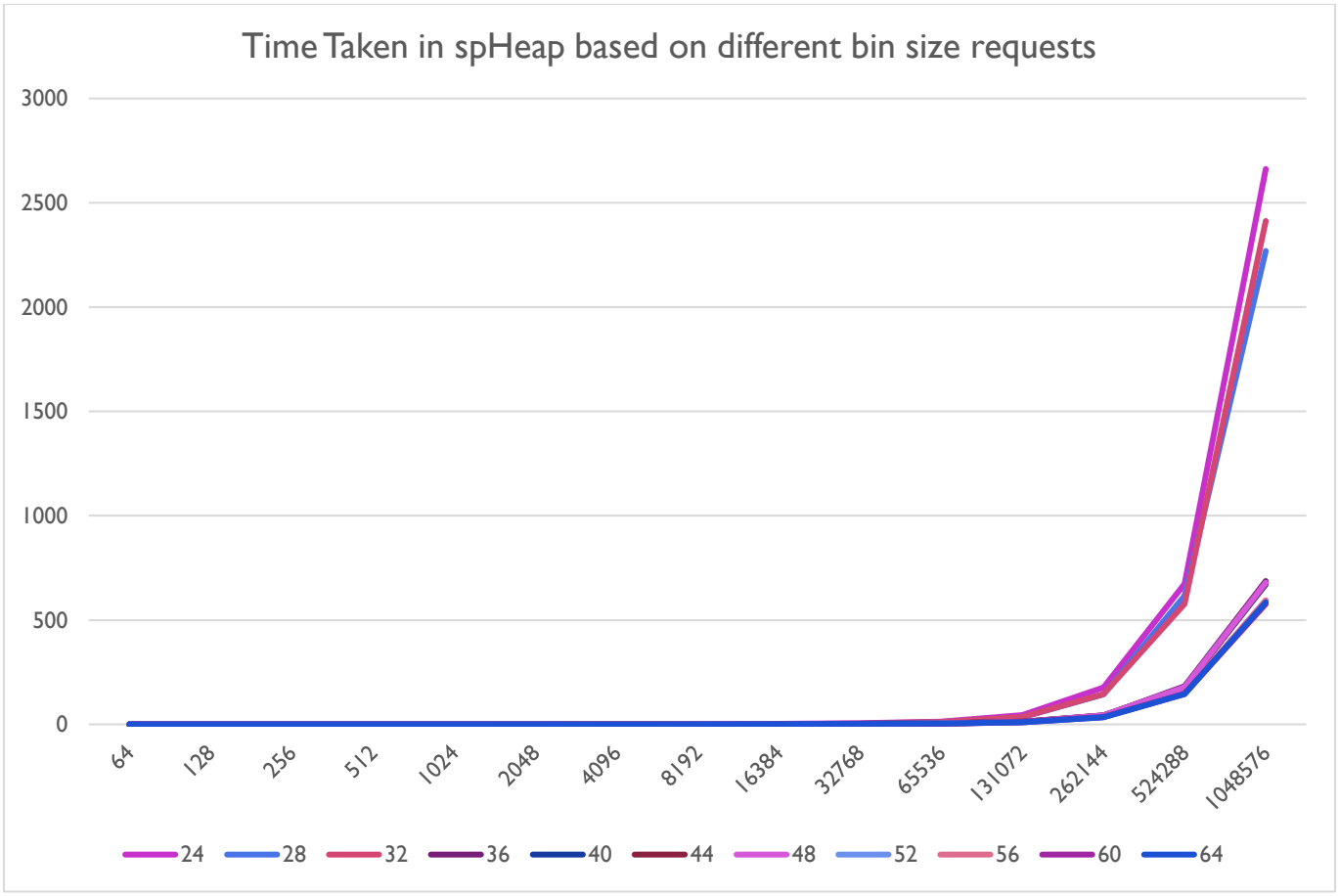
2. Performance: Report on Usage of OneBin vs Use of Weighted Memory Allocation

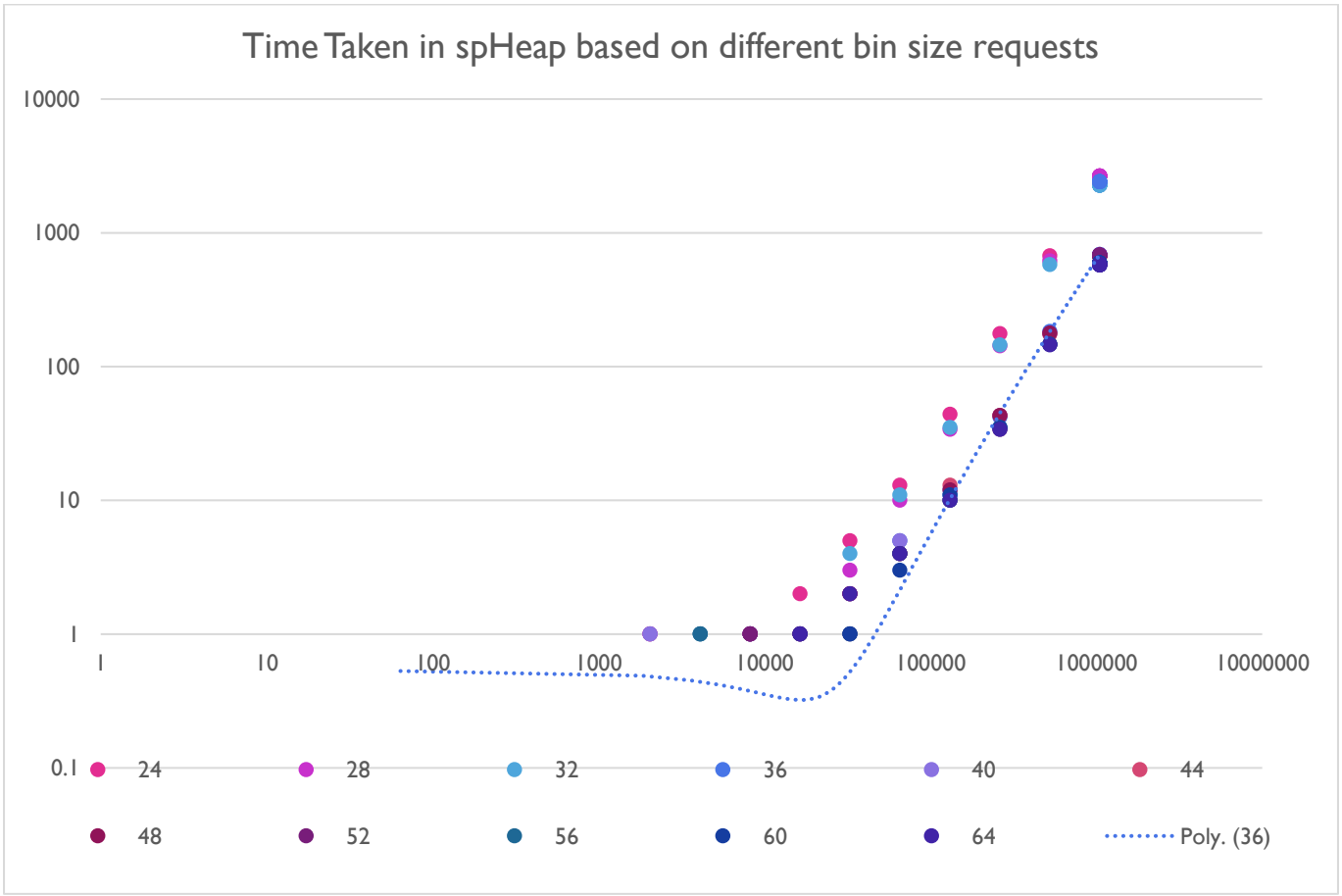
In order to get the worst case scenario, we allocated the memory heap till it was full, and then completely deallocated the memory until it was empty. The order in which the allocation and deallocation was done was FIFO to ensure, again, the worst case behaviour (LIFO is conducive to merges etc).

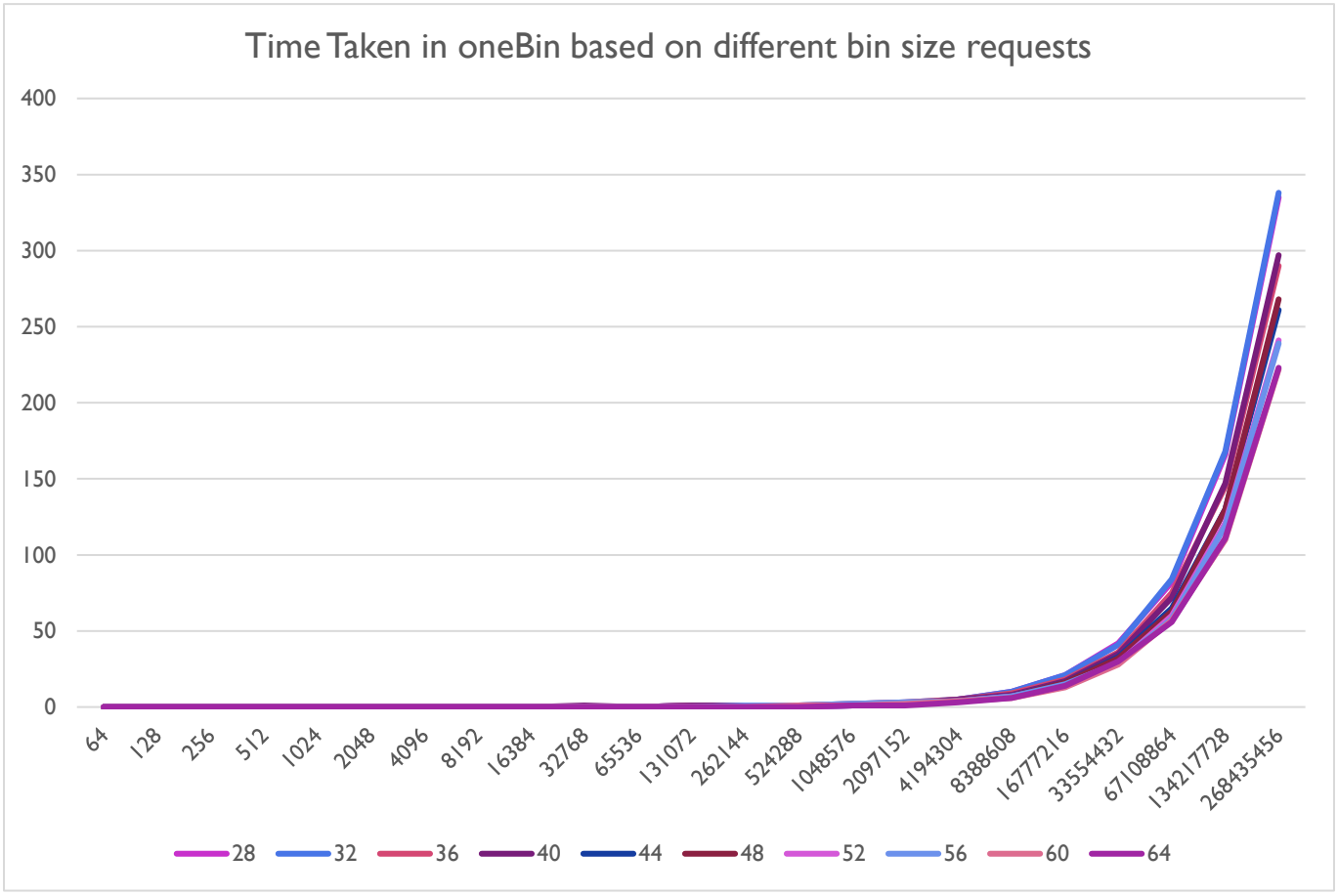
Conclusion: One bin is much more efficient than the weighted bin allocator for fixed memory sizes.

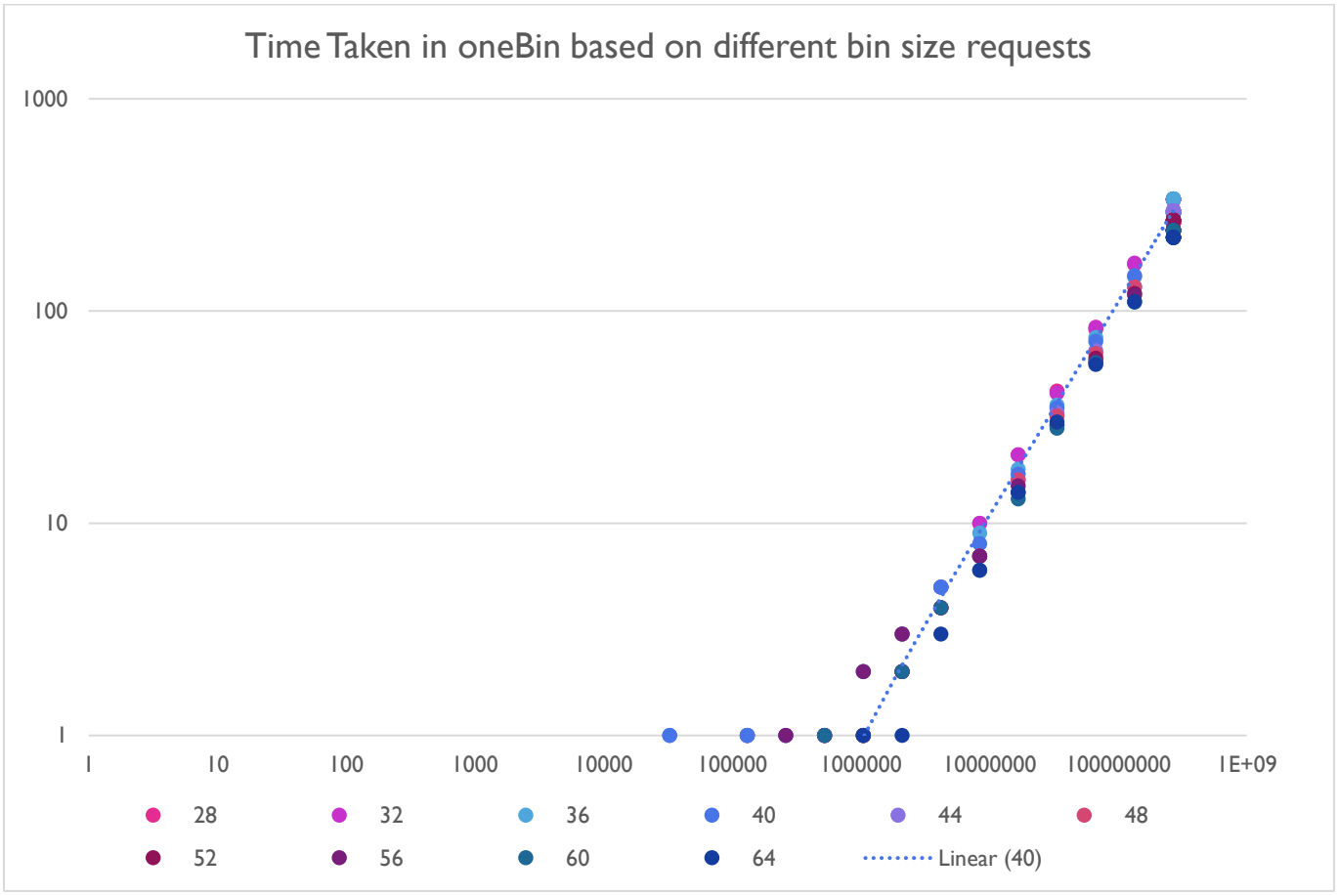





Time Taken in spHeap based on different bin size requests
3000
2500
2000
1500
1000
500
0
64
128
256
512
1024
2048
4096
8192
16384
32768
65536
131072
262144
524288
1048576
24
28
32
36
40
44
48
52
56
60
64


Time Taken in spHeap based on different bin size requests
10000
1000
100
10
1
0.1
1
10
100
1000
10000
100000
1000000
10000000
24
28
32
36
40
44
48
52
56
60
64
Poly. (36)


Time Taken in oneBin based on different bin size requests
400
350
300
250
200
150
100
50
0
64
128
256
512
1024
2048
4096
8192
16384
32768
65536
131072
262144
524288
1048576
2097152
4194304
8388608
16777216
33554432
67108864
134217728
268435456
28
32
36
40
44
48
52
56
60
64


Time Taken in oneBin based on different bin size requests
1000
100
10
1
1
10
100
1000
10000
100000
1000000
10000000
100000000
1E+09
28
32
36
40
44
48
52
56
60
64
Linear (40)

3 Tests on Polynomial Input

We ran the tests on polynomials, and to our surprise, oneBin was faster than malloc. The run on oneBin took just 2ms on average and the run on malloc took 4ms

The run on spHeap was nearly 100 times slower!

The actual outputs have been attached below in the appendix

Results for OneBin

```
Line 1697: Time taken in ms is 2
Line 3394: Time taken in ms is 3
Line 5091: Time taken in ms is 2
Line 6788: Time taken in ms is 3
Line 8485: Time taken in ms is 3
Line 10182: Time taken in ms is 2
Line 11879: Time taken in ms is 3
Line 13576: Time taken in ms is 3
Line 15273: Time taken in ms is 2
Line 16970: Time taken in ms is 3
```

Results for Malloc

```
Line 1687: Time taken in ms is 3
Line 3374: Time taken in ms is 4
Line 5061: Time taken in ms is 5
Line 6748: Time taken in ms is 4
Line 8435: Time taken in ms is 4
Line 10122: Time taken in ms is 5
Line 11809: Time taken in ms is 4
Line 13496: Time taken in ms is 4
Line 15183: Time taken in ms is 4
Line 16870: Time taken in ms is 4
```

Results for spHeap:

The result for spHeap was less promising at 131 ms

```
Line 1697: Time taken in ms is 198
Line 3394: Time taken in ms is 187
Line 5091: Time taken in ms is 196
Line 6788: Time taken in ms is 148
Line 8485: Time taken in ms is 125
Line 10182: Time taken in ms is 128
Line 11879: Time taken in ms is 125
Line 13576: Time taken in ms is 125
Line 15273: Time taken in ms is 126
Line 16970: Time taken in ms is 131
```

Here's the code that was run

1. test_polyListOps.c

```c
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include "common_header.h"

int main() {
    time_t timestart,timeend;
    timestart =clock();
    printf("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~\nTesting poly
Ops\n~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~\n");

    printf("\n## 1. Testing Generation, printing of polynomials and adding of monomials
to polynomials\n");
```

```
    for (int i = 0; i < 100; i+=10) {
        for (int j = 0; j < i&&j<10; j+=2) {
            printf("Printing a random %d degree polynomial with %d terms\n",i,j);
            poly*a = createRandomPoly(i,j,(unsigned int)time(0));
            print_Poly(a);
            freePoly(a);
        }
    }
    printf("\n## 2. Testing copyPoly\n");
    for (int i = 0; i < 100; i+=10) {
        for (int j = 0; j < i&&j<10; j+=2) {
            printf("Printing a random %d degree polynomial with %d terms\n",i,j);
            poly*a = createRandomPoly(i,j,(unsigned int)time(0));
            print_Poly(a);
            printf("Printing Copy of above\n");
            poly* b = copyPoly(a);
            print_Poly(b);
            freePoly(a);freePoly(b);
        }
    }

    printf("\n## 3. Testing mult_monomial_toPoly\n");
    for (int i = 0; i < 100; i+=10) {
        for (int j = 0; j < i&&j<10; j+=2) {
            printf("Printing a random %d degree polynomial with %d terms\n",i,j);
            poly*a = createRandomPoly(i,j,(unsigned int)time(0));
            print_Poly(a);
            float rand_coeff = (float)rand()/(float)(RAND_MAX/RAND_MAX_COEFF);
            unsigned int rand_mono_degree = (unsigned int) rand()%i;

            printf("Multiplying monomial above with coeff = %4f and power =
%d\n",rand_coeff,rand_mono_degree);
            poly* b = mult_monomial_toPoly(a,rand_coeff,rand_mono_degree);
            print_Poly(b);
            freePoly(a);freePoly(b);
        }
    }

    printf("\n## 4. Testing mult_monomial_toPoly_inplace\n");
    for (int i = 0; i < 100; i+=10) {
        for (int j = 0; j < i&&j<10; j+=2) {
            printf("Printing a random %d degree polynomial with %d terms\n",i,j);
            poly*a = createRandomPoly(i,j,(unsigned int)time(0));
            print_Poly(a);
            float rand_coeff = (float)rand()/(float)(RAND_MAX/RAND_MAX_COEFF);
            unsigned int rand_mono_degree = (unsigned int) rand()%i;

            printf("Multiplying monomial in place above with coeff = %4f and power =
%d\n",rand_coeff,rand_mono_degree);
            mult_monomial_toPoly_inplace(a,rand_coeff,rand_mono_degree);
            print_Poly(a);
            freePoly(a);
        }
    }

    printf("\n## 5. Testing addition of polynomial\n");
    for (int i = 0; i < 100; i+=10) {
        for (int j = 0; j < i&&j<10; j+=2) {
            printf("~~~~~~~~~~~~~~~~~~~~~\n");
            printf("Printing a random %d degree polynomial with %d terms\n",i,j);
            poly*a = createRandomPoly(i,j,(unsigned int)time(0));
            print_Poly(a);

            printf("Printing another random %d degree polynomial with %d terms\n",i,j);
            poly*b = createRandomPoly(i,j,25);
```

```
            print_Poly(b);

            printf("~~~~~PRINTING THEIR SUM~~~~~\n");

            poly* c = add_poly_toPoly(a,b);
            print_Poly(c);
            printf("~~~~~~~~~~~~~~~~~~~~~~\n");
            freePoly(a);freePoly(b);freePoly(c);
        }
    }
    printf("\n## 6. Testing subtraction of polynomials\n");
    for (int i = 0; i < 100; i+=10) {
        for (int j = 0; j < i&&j<10; j+=2) {
            printf("~~~~~~~~~~~~~~~~~~~~~~\n");
            printf("Printing a random %d degree polynomial A with %d terms\n",i,j);
            poly*a = createRandomPoly(i,j,(unsigned int)time(0));
            print_Poly(a);

            printf("Printing another random %d degree polynomial B with %d
terms\n",i,j);
            poly*b = createRandomPoly(i,j,25);
            print_Poly(b);

            printf("~~~~~PRINTING A-B~~~~~\n");

            poly* c = subtract_PolyB_from_PolyA(a,b);
            print_Poly(c);
            printf("~~~~~~~~~~~~~~~~~~~~~~\n");
            freePoly(a);freePoly(b);freePoly(c);
        }
    }
    printf("\n## 7. Testing division of polynomials\n");
    for (int i = 1; i < 100; i+=10) {
        for (int j = 1; j < i&&j<10; j+=2) {
            printf("~~~~~~~~~~~~~~~~~~~~~~\n");
            poly*a = createRandomPoly(i,j,(unsigned int)time(0));
            poly*b = createRandomPoly(i,j,25);

            printf("~~~~~PRINTING A/B~~~~~\n");
            poly* quotient = createEmptyPoly();
            poly* remainder = createEmptyPoly();
            divide_polyA_by_polyB(a,b,&quotient,&remainder);

printf("A\t=\t");print_Poly(a);printf("B\t=\t");print_Poly(b);printf("\tx\t");print_Poly
(quotient);printf("\t+\t");print_Poly(remainder);
            printf("~~~~~~~~~~~~~~~~~~~~~~\n");
            freePoly(a);freePoly(b);freePoly(quotient);freePoly(remainder);
        }
    }



    printf("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~\nTESTING
COMPLETED\n~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~\n~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~\n");
    timeend = clock();
    printf("Time taken in ms is %d",(int)(timeend-timestart));
    return 0;
}
```

4.1. Experiment from spHeap $O(n^2)$

The Experiment Statistics are as follows
The Average Internal Fragmentation was 10.43%
The Average External Fragmentation was 43.85%
The Average Time Taken in ms. per MB was 240.26

```
------------------------------------------------------------------------
------------------------------------------------------------------------
      Starting the experiment for spHeap
------------------------------------------------------------------------
------------------------------------------------------------------------
   MemSize ||    Bin   || Cnt    || IntFrag|| ExtFrag    || TimeTaken
      32B  ||     24   ||   1    ||  0.00% ||  100.00 %  || 0ms
      32B  ||     24   ||   2    ||  0.00% ||  100.00 %  || 0ms
      32B  ||     24   ||   3    ||  0.00% ||  100.00 %  || 0ms
      32B  ||     24   ||   4    ||  0.00% ||  100.00 %  || 0ms
      32B  ||     24   ||   5    ||  0.00% ||  100.00 %  || 0ms
      32B  ||     28   ||   1    ||  0.00% ||  100.00 %  || 0ms
      32B  ||     28   ||   2    ||  0.00% ||  100.00 %  || 0ms
      32B  ||     28   ||   3    ||  0.00% ||  100.00 %  || 0ms
      32B  ||     28   ||   4    ||  0.00% ||  100.00 %  || 0ms
      32B  ||     28   ||   5    ||  0.00% ||  100.00 %  || 0ms
      32B  ||     32   ||   1    ||  0.00% ||  100.00 %  || 0ms
      32B  ||     32   ||   2    ||  0.00% ||  100.00 %  || 0ms
      32B  ||     32   ||   3    ||  0.00% ||  100.00 %  || 0ms
      32B  ||     32   ||   4    ||  0.00% ||  100.00 %  || 0ms
      32B  ||     32   ||   5    ||  0.00% ||  100.00 %  || 0ms
      64B  ||     24   ||   1    ||  0.00% ||  62.50 %   || 0ms
      64B  ||     24   ||   2    ||  0.00% ||  62.50 %   || 0ms
      64B  ||     24   ||   3    ||  0.00% ||  62.50 %   || 0ms
      64B  ||     24   ||   4    ||  0.00% ||  62.50 %   || 0ms
      64B  ||     24   ||   5    ||  0.00% ||  62.50 %   || 0ms
      64B  ||     28   ||   1    || 14.29% ||  50.00 %   || 0ms
      64B  ||     28   ||   2    || 14.29% ||  50.00 %   || 0ms
      64B  ||     28   ||   3    || 14.29% ||  50.00 %   || 0ms
      64B  ||     28   ||   4    || 14.29% ||  50.00 %   || 0ms
      64B  ||     28   ||   5    || 14.29% ||  50.00 %   || 0ms
      64B  ||     32   ||   1    ||  0.00% ||  50.00 %   || 0ms
      64B  ||     32   ||   2    ||  0.00% ||  50.00 %   || 0ms
      64B  ||     32   ||   3    ||  0.00% ||  50.00 %   || 0ms
      64B  ||     32   ||   4    ||  0.00% ||  50.00 %   || 0ms
      64B  ||     32   ||   5    ||  0.00% ||  50.00 %   || 0ms
      64B  ||     36   ||   1    ||  0.00% ||  100.00 %  || 0ms
      64B  ||     36   ||   2    ||  0.00% ||  100.00 %  || 0ms
      64B  ||     36   ||   3    ||  0.00% ||  100.00 %  || 0ms
      64B  ||     36   ||   4    ||  0.00% ||  100.00 %  || 0ms
      64B  ||     36   ||   5    ||  0.00% ||  100.00 %  || 0ms
      64B  ||     40   ||   1    ||  0.00% ||  100.00 %  || 0ms
      64B  ||     40   ||   2    ||  0.00% ||  100.00 %  || 0ms
      64B  ||     40   ||   3    ||  0.00% ||  100.00 %  || 0ms
      64B  ||     40   ||   4    ||  0.00% ||  100.00 %  || 0ms
      64B  ||     40   ||   5    ||  0.00% ||  100.00 %  || 0ms
      64B  ||     44   ||   1    ||  0.00% ||  100.00 %  || 0ms
      64B  ||     44   ||   2    ||  0.00% ||  100.00 %  || 0ms
      64B  ||     44   ||   3    ||  0.00% ||  100.00 %  || 0ms
      64B  ||     44   ||   4    ||  0.00% ||  100.00 %  || 0ms
      64B  ||     44   ||   5    ||  0.00% ||  100.00 %  || 0ms
      64B  ||     48   ||   1    ||  0.00% ||  100.00 %  || 0ms
      64B  ||     48   ||   2    ||  0.00% ||  100.00 %  || 0ms
      64B  ||     48   ||   3    ||  0.00% ||  100.00 %  || 0ms
      64B  ||     48   ||   4    ||  0.00% ||  100.00 %  || 0ms
      64B  ||     48   ||   5    ||  0.00% ||  100.00 %  || 0ms
      64B  ||     52   ||   1    ||  0.00% ||  100.00 %  || 0ms
      64B  ||     52   ||   2    ||  0.00% ||  100.00 %  || 0ms
```

64B	52	3	0.00%	100.00 %	0ms
64B	52	4	0.00%	100.00 %	0ms
64B	52	5	0.00%	100.00 %	0ms
64B	56	1	0.00%	100.00 %	0ms
64B	56	2	0.00%	100.00 %	0ms
64B	56	3	0.00%	100.00 %	0ms
64B	56	4	0.00%	100.00 %	0ms
64B	56	5	0.00%	100.00 %	0ms
64B	60	1	0.00%	100.00 %	0ms
64B	60	2	0.00%	100.00 %	0ms
64B	60	3	0.00%	100.00 %	0ms
64B	60	4	0.00%	100.00 %	0ms
64B	60	5	0.00%	100.00 %	0ms
64B	64	1	0.00%	100.00 %	0ms
64B	64	2	0.00%	100.00 %	0ms
64B	64	3	0.00%	100.00 %	0ms
64B	64	4	0.00%	100.00 %	0ms
64B	64	5	0.00%	100.00 %	0ms
128B	24	1	0.00%	43.75 %	0ms
128B	24	2	0.00%	43.75 %	0ms
128B	24	3	0.00%	43.75 %	0ms
128B	24	4	0.00%	43.75 %	0ms
128B	24	5	0.00%	43.75 %	0ms
128B	28	1	14.29%	25.00 %	0ms
128B	28	2	14.29%	25.00 %	0ms
128B	28	3	14.29%	25.00 %	0ms
128B	28	4	14.29%	25.00 %	0ms
128B	28	5	14.29%	25.00 %	0ms
128B	32	1	0.00%	25.00 %	0ms
128B	32	2	0.00%	25.00 %	0ms
128B	32	3	0.00%	25.00 %	0ms
128B	32	4	0.00%	25.00 %	0ms
128B	32	5	0.00%	25.00 %	0ms
128B	36	1	33.33%	62.50 %	0ms
128B	36	2	33.33%	62.50 %	0ms
128B	36	3	33.33%	62.50 %	0ms
128B	36	4	33.33%	62.50 %	0ms
128B	36	5	33.33%	62.50 %	0ms
128B	40	1	20.00%	62.50 %	0ms
128B	40	2	20.00%	62.50 %	0ms
128B	40	3	20.00%	62.50 %	0ms
128B	40	4	20.00%	62.50 %	0ms
128B	40	5	20.00%	62.50 %	0ms
128B	44	1	9.09%	62.50 %	0ms
128B	44	2	9.09%	62.50 %	0ms
128B	44	3	9.09%	62.50 %	0ms
128B	44	4	9.09%	62.50 %	0ms
128B	44	5	9.09%	62.50 %	0ms
128B	48	1	0.00%	62.50 %	0ms
128B	48	2	0.00%	62.50 %	0ms
128B	48	3	0.00%	62.50 %	0ms
128B	48	4	0.00%	62.50 %	0ms
128B	48	5	0.00%	62.50 %	0ms
128B	52	1	23.08%	50.00 %	0ms
128B	52	2	23.08%	50.00 %	0ms
128B	52	3	23.08%	50.00 %	0ms
128B	52	4	23.08%	50.00 %	0ms
128B	52	5	23.08%	50.00 %	0ms
128B	56	1	14.29%	50.00 %	0ms
128B	56	2	14.29%	50.00 %	0ms
128B	56	3	14.29%	50.00 %	0ms
128B	56	4	14.29%	50.00 %	0ms
128B	56	5	14.29%	50.00 %	0ms
128B	60	1	6.67%	50.00 %	0ms
128B	60	2	6.67%	50.00 %	0ms

128B	60	3	6.67%	50.00 %	0ms
128B	60	4	6.67%	50.00 %	0ms
128B	60	5	6.67%	50.00 %	0ms
128B	64	1	0.00%	50.00 %	0ms
128B	64	2	0.00%	50.00 %	0ms
128B	64	3	0.00%	50.00 %	0ms
128B	64	4	0.00%	50.00 %	0ms
128B	64	5	0.00%	50.00 %	0ms
256B	24	1	0.00%	53.13 %	0ms
256B	24	2	0.00%	53.13 %	0ms
256B	24	3	0.00%	53.13 %	0ms
256B	24	4	0.00%	53.13 %	0ms
256B	24	5	0.00%	53.13 %	0ms
256B	28	1	14.29%	37.50 %	0ms
256B	28	2	14.29%	37.50 %	0ms
256B	28	3	14.29%	37.50 %	0ms
256B	28	4	14.29%	37.50 %	0ms
256B	28	5	14.29%	37.50 %	0ms
256B	32	1	0.00%	37.50 %	0ms
256B	32	2	0.00%	37.50 %	0ms
256B	32	3	0.00%	37.50 %	0ms
256B	32	4	0.00%	37.50 %	0ms
256B	32	5	0.00%	37.50 %	0ms
256B	36	1	33.33%	43.75 %	0ms
256B	36	2	33.33%	43.75 %	0ms
256B	36	3	33.33%	43.75 %	0ms
256B	36	4	33.33%	43.75 %	0ms
256B	36	5	33.33%	43.75 %	0ms
256B	40	1	20.00%	43.75 %	0ms
256B	40	2	20.00%	43.75 %	0ms
256B	40	3	20.00%	43.75 %	0ms
256B	40	4	20.00%	43.75 %	0ms
256B	40	5	20.00%	43.75 %	0ms
256B	44	1	9.09%	43.75 %	0ms
256B	44	2	9.09%	43.75 %	0ms
256B	44	3	9.09%	43.75 %	0ms
256B	44	4	9.09%	43.75 %	0ms
256B	44	5	9.09%	43.75 %	0ms
256B	48	1	0.00%	43.75 %	0ms
256B	48	2	0.00%	43.75 %	0ms
256B	48	3	0.00%	43.75 %	0ms
256B	48	4	0.00%	43.75 %	0ms
256B	48	5	0.00%	43.75 %	0ms
256B	52	1	23.08%	25.00 %	0ms
256B	52	2	23.08%	25.00 %	0ms
256B	52	3	23.08%	25.00 %	0ms
256B	52	4	23.08%	25.00 %	0ms
256B	52	5	23.08%	25.00 %	0ms
256B	56	1	14.29%	25.00 %	0ms
256B	56	2	14.29%	25.00 %	0ms
256B	56	3	14.29%	25.00 %	0ms
256B	56	4	14.29%	25.00 %	0ms
256B	56	5	14.29%	25.00 %	0ms
256B	60	1	6.67%	25.00 %	0ms
256B	60	2	6.67%	25.00 %	0ms
256B	60	3	6.67%	25.00 %	0ms
256B	60	4	6.67%	25.00 %	0ms
256B	60	5	6.67%	25.00 %	0ms
256B	64	1	0.00%	25.00 %	0ms
256B	64	2	0.00%	25.00 %	0ms
256B	64	3	0.00%	25.00 %	0ms
256B	64	4	0.00%	25.00 %	0ms
256B	64	5	0.00%	25.00 %	0ms
512B	24	1	0.00%	48.44 %	1ms
512B	24	2	0.00%	48.44 %	0ms

512B	24	3	0.00%	48.44 %	0ms
512B	24	4	0.00%	48.44 %	0ms
512B	24	5	0.00%	48.44 %	0ms
512B	28	1	14.29%	31.25 %	0ms
512B	28	2	14.29%	31.25 %	0ms
512B	28	3	14.29%	31.25 %	0ms
512B	28	4	14.29%	31.25 %	0ms
512B	28	5	14.29%	31.25 %	0ms
512B	32	1	0.00%	31.25 %	0ms
512B	32	2	0.00%	31.25 %	0ms
512B	32	3	0.00%	31.25 %	0ms
512B	32	4	0.00%	31.25 %	0ms
512B	32	5	0.00%	31.25 %	0ms
512B	36	1	33.33%	53.13 %	0ms
512B	36	2	33.33%	53.13 %	0ms
512B	36	3	33.33%	53.13 %	0ms
512B	36	4	33.33%	53.13 %	0ms
512B	36	5	33.33%	53.13 %	0ms
512B	40	1	20.00%	53.13 %	0ms
512B	40	2	20.00%	53.13 %	0ms
512B	40	3	20.00%	53.13 %	0ms
512B	40	4	20.00%	53.13 %	0ms
512B	40	5	20.00%	53.13 %	0ms
512B	44	1	9.09%	53.13 %	1ms
512B	44	2	9.09%	53.13 %	0ms
512B	44	3	9.09%	53.13 %	0ms
512B	44	4	9.09%	53.13 %	0ms
512B	44	5	9.09%	53.13 %	0ms
512B	48	1	0.00%	53.13 %	0ms
512B	48	2	0.00%	53.13 %	0ms
512B	48	3	0.00%	53.13 %	0ms
512B	48	4	0.00%	53.13 %	0ms
512B	48	5	0.00%	53.13 %	0ms
512B	52	1	23.08%	37.50 %	0ms
512B	52	2	23.08%	37.50 %	0ms
512B	52	3	23.08%	37.50 %	0ms
512B	52	4	23.08%	37.50 %	0ms
512B	52	5	23.08%	37.50 %	0ms
512B	56	1	14.29%	37.50 %	0ms
512B	56	2	14.29%	37.50 %	0ms
512B	56	3	14.29%	37.50 %	0ms
512B	56	4	14.29%	37.50 %	0ms
512B	56	5	14.29%	37.50 %	0ms
512B	60	1	6.67%	37.50 %	0ms
512B	60	2	6.67%	37.50 %	0ms
512B	60	3	6.67%	37.50 %	0ms
512B	60	4	6.67%	37.50 %	0ms
512B	60	5	6.67%	37.50 %	0ms
512B	64	1	0.00%	37.50 %	0ms
512B	64	2	0.00%	37.50 %	0ms
512B	64	3	0.00%	37.50 %	0ms
512B	64	4	0.00%	37.50 %	0ms
512B	64	5	0.00%	37.50 %	0ms
1024B	24	1	0.00%	50.78 %	0ms
1024B	24	2	0.00%	50.78 %	0ms
1024B	24	3	0.00%	50.78 %	1ms
1024B	24	4	0.00%	50.78 %	0ms
1024B	24	5	0.00%	50.78 %	0ms
1024B	28	1	14.29%	34.38 %	0ms
1024B	28	2	14.29%	34.38 %	0ms
1024B	28	3	14.29%	34.38 %	0ms
1024B	28	4	14.29%	34.38 %	0ms
1024B	28	5	14.29%	34.38 %	0ms
1024B	32	1	0.00%	34.38 %	0ms
1024B	32	2	0.00%	34.38 %	0ms

1024B	32	3	0.00%	34.38 %	0ms
1024B	32	4	0.00%	34.38 %	1ms
1024B	32	5	0.00%	34.38 %	0ms
1024B	36	1	33.33%	48.44 %	0ms
1024B	36	2	33.33%	48.44 %	0ms
1024B	36	3	33.33%	48.44 %	0ms
1024B	36	4	33.33%	48.44 %	0ms
1024B	36	5	33.33%	48.44 %	0ms
1024B	40	1	20.00%	48.44 %	0ms
1024B	40	2	20.00%	48.44 %	0ms
1024B	40	3	20.00%	48.44 %	0ms
1024B	40	4	20.00%	48.44 %	0ms
1024B	40	5	20.00%	48.44 %	0ms
1024B	44	1	9.09%	48.44 %	0ms
1024B	44	2	9.09%	48.44 %	0ms
1024B	44	3	9.09%	48.44 %	0ms
1024B	44	4	9.09%	48.44 %	0ms
1024B	44	5	9.09%	48.44 %	0ms
1024B	48	1	0.00%	48.44 %	0ms
1024B	48	2	0.00%	48.44 %	0ms
1024B	48	3	0.00%	48.44 %	1ms
1024B	48	4	0.00%	48.44 %	0ms
1024B	48	5	0.00%	48.44 %	0ms
1024B	52	1	23.08%	31.25 %	0ms
1024B	52	2	23.08%	31.25 %	0ms
1024B	52	3	23.08%	31.25 %	0ms
1024B	52	4	23.08%	31.25 %	0ms
1024B	52	5	23.08%	31.25 %	0ms
1024B	56	1	14.29%	31.25 %	0ms
1024B	56	2	14.29%	31.25 %	0ms
1024B	56	3	14.29%	31.25 %	0ms
1024B	56	4	14.29%	31.25 %	0ms
1024B	56	5	14.29%	31.25 %	0ms
1024B	60	1	6.67%	31.25 %	0ms
1024B	60	2	6.67%	31.25 %	0ms
1024B	60	3	6.67%	31.25 %	0ms
1024B	60	4	6.67%	31.25 %	0ms
1024B	60	5	6.67%	31.25 %	0ms
1024B	64	1	0.00%	31.25 %	0ms
1024B	64	2	0.00%	31.25 %	0ms
1024B	64	3	0.00%	31.25 %	0ms
1024B	64	4	0.00%	31.25 %	0ms
1024B	64	5	0.00%	31.25 %	0ms
2.00KB	24	1	0.00%	49.61 %	1ms
2.00KB	24	2	0.00%	49.61 %	0ms
2.00KB	24	3	0.00%	49.61 %	0ms
2.00KB	24	4	0.00%	49.61 %	0ms
2.00KB	24	5	0.00%	49.61 %	0ms
2.00KB	28	1	14.29%	32.81 %	1ms
2.00KB	28	2	14.29%	32.81 %	0ms
2.00KB	28	3	14.29%	32.81 %	0ms
2.00KB	28	4	14.29%	32.81 %	0ms
2.00KB	28	5	14.29%	32.81 %	0ms
2.00KB	32	1	0.00%	32.81 %	0ms
2.00KB	32	2	0.00%	32.81 %	0ms
2.00KB	32	3	0.00%	32.81 %	1ms
2.00KB	32	4	0.00%	32.81 %	0ms
2.00KB	32	5	0.00%	32.81 %	0ms
2.00KB	36	1	33.33%	50.78 %	0ms
2.00KB	36	2	33.33%	50.78 %	0ms
2.00KB	36	3	33.33%	50.78 %	0ms
2.00KB	36	4	33.33%	50.78 %	0ms
2.00KB	36	5	33.33%	50.78 %	0ms
2.00KB	40	1	20.00%	50.78 %	0ms
2.00KB	40	2	20.00%	50.78 %	1ms

2.00KB	40	3	20.00%	50.78 %	0ms
2.00KB	40	4	20.00%	50.78 %	0ms
2.00KB	40	5	20.00%	50.78 %	0ms
2.00KB	44	1	9.09%	50.78 %	0ms
2.00KB	44	2	9.09%	50.78 %	0ms
2.00KB	44	3	9.09%	50.78 %	0ms
2.00KB	44	4	9.09%	50.78 %	0ms
2.00KB	44	5	9.09%	50.78 %	0ms
2.00KB	48	1	0.00%	50.78 %	1ms
2.00KB	48	2	0.00%	50.78 %	0ms
2.00KB	48	3	0.00%	50.78 %	0ms
2.00KB	48	4	0.00%	50.78 %	0ms
2.00KB	48	5	0.00%	50.78 %	0ms
2.00KB	52	1	23.08%	34.38 %	0ms
2.00KB	52	2	23.08%	34.38 %	0ms
2.00KB	52	3	23.08%	34.38 %	0ms
2.00KB	52	4	23.08%	34.38 %	0ms
2.00KB	52	5	23.08%	34.38 %	0ms
2.00KB	56	1	14.29%	34.38 %	0ms
2.00KB	56	2	14.29%	34.38 %	0ms
2.00KB	56	3	14.29%	34.38 %	1ms
2.00KB	56	4	14.29%	34.38 %	0ms
2.00KB	56	5	14.29%	34.38 %	0ms
2.00KB	60	1	6.67%	34.38 %	0ms
2.00KB	60	2	6.67%	34.38 %	0ms
2.00KB	60	3	6.67%	34.38 %	0ms
2.00KB	60	4	6.67%	34.38 %	0ms
2.00KB	60	5	6.67%	34.38 %	0ms
2.00KB	64	1	0.00%	34.38 %	0ms
2.00KB	64	2	0.00%	34.38 %	0ms
2.00KB	64	3	0.00%	34.38 %	0ms
2.00KB	64	4	0.00%	34.38 %	0ms
2.00KB	64	5	0.00%	34.38 %	0ms
4.00KB	24	1	0.00%	50.20 %	1ms
4.00KB	24	2	0.00%	50.20 %	0ms
4.00KB	24	3	0.00%	50.20 %	1ms
4.00KB	24	4	0.00%	50.20 %	0ms
4.00KB	24	5	0.00%	50.20 %	1ms
4.00KB	28	1	14.29%	33.59 %	0ms
4.00KB	28	2	14.29%	33.59 %	0ms
4.00KB	28	3	14.29%	33.59 %	0ms
4.00KB	28	4	14.29%	33.59 %	1ms
4.00KB	28	5	14.29%	33.59 %	0ms
4.00KB	32	1	0.00%	33.59 %	0ms
4.00KB	32	2	0.00%	33.59 %	1ms
4.00KB	32	3	0.00%	33.59 %	0ms
4.00KB	32	4	0.00%	33.59 %	0ms
4.00KB	32	5	0.00%	33.59 %	0ms
4.00KB	36	1	33.33%	49.61 %	1ms
4.00KB	36	2	33.33%	49.61 %	0ms
4.00KB	36	3	33.33%	49.61 %	0ms
4.00KB	36	4	33.33%	49.61 %	0ms
4.00KB	36	5	33.33%	49.61 %	0ms
4.00KB	40	1	20.00%	49.61 %	1ms
4.00KB	40	2	20.00%	49.61 %	0ms
4.00KB	40	3	20.00%	49.61 %	0ms
4.00KB	40	4	20.00%	49.61 %	0ms
4.00KB	40	5	20.00%	49.61 %	1ms
4.00KB	44	1	9.09%	49.61 %	0ms
4.00KB	44	2	9.09%	49.61 %	0ms
4.00KB	44	3	9.09%	49.61 %	0ms
4.00KB	44	4	9.09%	49.61 %	0ms
4.00KB	44	5	9.09%	49.61 %	1ms
4.00KB	48	1	0.00%	49.61 %	0ms
4.00KB	48	2	0.00%	49.61 %	0ms

4.00KB	48	3	0.00%	49.61 %	0ms
4.00KB	48	4	0.00%	49.61 %	0ms
4.00KB	48	5	0.00%	49.61 %	1ms
4.00KB	52	1	23.08%	32.81 %	0ms
4.00KB	52	2	23.08%	32.81 %	0ms
4.00KB	52	3	23.08%	32.81 %	0ms
4.00KB	52	4	23.08%	32.81 %	0ms
4.00KB	52	5	23.08%	32.81 %	0ms
4.00KB	56	1	14.29%	32.81 %	0ms
4.00KB	56	2	14.29%	32.81 %	1ms
4.00KB	56	3	14.29%	32.81 %	0ms
4.00KB	56	4	14.29%	32.81 %	0ms
4.00KB	56	5	14.29%	32.81 %	0ms
4.00KB	60	1	6.67%	32.81 %	0ms
4.00KB	60	2	6.67%	32.81 %	0ms
4.00KB	60	3	6.67%	32.81 %	1ms
4.00KB	60	4	6.67%	32.81 %	0ms
4.00KB	60	5	6.67%	32.81 %	0ms
4.00KB	64	1	0.00%	32.81 %	0ms
4.00KB	64	2	0.00%	32.81 %	0ms
4.00KB	64	3	0.00%	32.81 %	0ms
4.00KB	64	4	0.00%	32.81 %	0ms
4.00KB	64	5	0.00%	32.81 %	1ms
8.00KB	24	1	0.00%	49.90 %	0ms
8.00KB	24	2	0.00%	49.90 %	1ms
8.00KB	24	3	0.00%	49.90 %	1ms
8.00KB	24	4	0.00%	49.90 %	1ms
8.00KB	24	5	0.00%	49.90 %	1ms
8.00KB	28	1	14.29%	33.20 %	0ms
8.00KB	28	2	14.29%	33.20 %	1ms
8.00KB	28	3	14.29%	33.20 %	1ms
8.00KB	28	4	14.29%	33.20 %	0ms
8.00KB	28	5	14.29%	33.20 %	1ms
8.00KB	32	1	0.00%	33.20 %	1ms
8.00KB	32	2	0.00%	33.20 %	0ms
8.00KB	32	3	0.00%	33.20 %	1ms
8.00KB	32	4	0.00%	33.20 %	0ms
8.00KB	32	5	0.00%	33.20 %	1ms
8.00KB	36	1	33.33%	50.20 %	0ms
8.00KB	36	2	33.33%	50.20 %	1ms
8.00KB	36	3	33.33%	50.20 %	0ms
8.00KB	36	4	33.33%	50.20 %	1ms
8.00KB	36	5	33.33%	50.20 %	0ms
8.00KB	40	1	20.00%	50.20 %	1ms
8.00KB	40	2	20.00%	50.20 %	0ms
8.00KB	40	3	20.00%	50.20 %	0ms
8.00KB	40	4	20.00%	50.20 %	1ms
8.00KB	40	5	20.00%	50.20 %	0ms
8.00KB	44	1	9.09%	50.20 %	1ms
8.00KB	44	2	9.09%	50.20 %	0ms
8.00KB	44	3	9.09%	50.20 %	1ms
8.00KB	44	4	9.09%	50.20 %	0ms
8.00KB	44	5	9.09%	50.20 %	0ms
8.00KB	48	1	0.00%	50.20 %	1ms
8.00KB	48	2	0.00%	50.20 %	0ms
8.00KB	48	3	0.00%	50.20 %	1ms
8.00KB	48	4	0.00%	50.20 %	0ms
8.00KB	48	5	0.00%	50.20 %	0ms
8.00KB	52	1	23.08%	33.59 %	1ms
8.00KB	52	2	23.08%	33.59 %	0ms
8.00KB	52	3	23.08%	33.59 %	0ms
8.00KB	52	4	23.08%	33.59 %	1ms
8.00KB	52	5	23.08%	33.59 %	0ms
8.00KB	56	1	14.29%	33.59 %	0ms
8.00KB	56	2	14.29%	33.59 %	1ms

8.00KB	56	3	14.29%	33.59 %	0ms
8.00KB	56	4	14.29%	33.59 %	0ms
8.00KB	56	5	14.29%	33.59 %	0ms
8.00KB	60	1	6.67%	33.59 %	1ms
8.00KB	60	2	6.67%	33.59 %	0ms
8.00KB	60	3	6.67%	33.59 %	0ms
8.00KB	60	4	6.67%	33.59 %	1ms
8.00KB	60	5	6.67%	33.59 %	0ms
8.00KB	64	1	0.00%	33.59 %	0ms
8.00KB	64	2	0.00%	33.59 %	0ms
8.00KB	64	3	0.00%	33.59 %	1ms
8.00KB	64	4	0.00%	33.59 %	0ms
8.00KB	64	5	0.00%	33.59 %	0ms
16.00KB	24	1	0.00%	50.05 %	2ms
16.00KB	24	2	0.00%	50.05 %	2ms
16.00KB	24	3	0.00%	50.05 %	2ms
16.00KB	24	4	0.00%	50.05 %	2ms
16.00KB	24	5	0.00%	50.05 %	2ms
16.00KB	28	1	14.29%	33.40 %	1ms
16.00KB	28	2	14.29%	33.40 %	1ms
16.00KB	28	3	14.29%	33.40 %	2ms
16.00KB	28	4	14.29%	33.40 %	1ms
16.00KB	28	5	14.29%	33.40 %	2ms
16.00KB	32	1	0.00%	33.40 %	1ms
16.00KB	32	2	0.00%	33.40 %	1ms
16.00KB	32	3	0.00%	33.40 %	2ms
16.00KB	32	4	0.00%	33.40 %	1ms
16.00KB	32	5	0.00%	33.40 %	1ms
16.00KB	36	1	33.33%	49.90 %	1ms
16.00KB	36	2	33.33%	49.90 %	1ms
16.00KB	36	3	33.33%	49.90 %	1ms
16.00KB	36	4	33.33%	49.90 %	1ms
16.00KB	36	5	33.33%	49.90 %	1ms
16.00KB	40	1	20.00%	49.90 %	1ms
16.00KB	40	2	20.00%	49.90 %	0ms
16.00KB	40	3	20.00%	49.90 %	1ms
16.00KB	40	4	20.00%	49.90 %	1ms
16.00KB	40	5	20.00%	49.90 %	1ms
16.00KB	44	1	9.09%	49.90 %	1ms
16.00KB	44	2	9.09%	49.90 %	1ms
16.00KB	44	3	9.09%	49.90 %	1ms
16.00KB	44	4	9.09%	49.90 %	0ms
16.00KB	44	5	9.09%	49.90 %	1ms
16.00KB	48	1	0.00%	49.90 %	1ms
16.00KB	48	2	0.00%	49.90 %	1ms
16.00KB	48	3	0.00%	49.90 %	1ms
16.00KB	48	4	0.00%	49.90 %	1ms
16.00KB	48	5	0.00%	49.90 %	1ms
16.00KB	52	1	23.08%	33.20 %	0ms
16.00KB	52	2	23.08%	33.20 %	1ms
16.00KB	52	3	23.08%	33.20 %	1ms
16.00KB	52	4	23.08%	33.20 %	0ms
16.00KB	52	5	23.08%	33.20 %	1ms
16.00KB	56	1	14.29%	33.20 %	0ms
16.00KB	56	2	14.29%	33.20 %	1ms
16.00KB	56	3	14.29%	33.20 %	1ms
16.00KB	56	4	14.29%	33.20 %	0ms
16.00KB	56	5	14.29%	33.20 %	1ms
16.00KB	60	1	6.67%	33.20 %	1ms
16.00KB	60	2	6.67%	33.20 %	0ms
16.00KB	60	3	6.67%	33.20 %	1ms
16.00KB	60	4	6.67%	33.20 %	0ms
16.00KB	60	5	6.67%	33.20 %	1ms
16.00KB	64	1	0.00%	33.20 %	1ms
16.00KB	64	2	0.00%	33.20 %	0ms

16.00KB	64	3	0.00%	33.20 %	1ms
16.00KB	64	4	0.00%	33.20 %	0ms
16.00KB	64	5	0.00%	33.20 %	1ms
32.00KB	24	1	0.00%	49.98 %	5ms
32.00KB	24	2	0.00%	49.98 %	4ms
32.00KB	24	3	0.00%	49.98 %	5ms
32.00KB	24	4	0.00%	49.98 %	4ms
32.00KB	24	5	0.00%	49.98 %	5ms
32.00KB	28	1	14.29%	33.30 %	3ms
32.00KB	28	2	14.29%	33.30 %	4ms
32.00KB	28	3	14.29%	33.30 %	3ms
32.00KB	28	4	14.29%	33.30 %	4ms
32.00KB	28	5	14.29%	33.30 %	3ms
32.00KB	32	1	0.00%	33.30 %	4ms
32.00KB	32	2	0.00%	33.30 %	3ms
32.00KB	32	3	0.00%	33.30 %	4ms
32.00KB	32	4	0.00%	33.30 %	3ms
32.00KB	32	5	0.00%	33.30 %	4ms
32.00KB	36	1	33.33%	50.05 %	1ms
32.00KB	36	2	33.33%	50.05 %	2ms
32.00KB	36	3	33.33%	50.05 %	2ms
32.00KB	36	4	33.33%	50.05 %	2ms
32.00KB	36	5	33.33%	50.05 %	2ms
32.00KB	40	1	20.00%	50.05 %	2ms
32.00KB	40	2	20.00%	50.05 %	2ms
32.00KB	40	3	20.00%	50.05 %	1ms
32.00KB	40	4	20.00%	50.05 %	2ms
32.00KB	40	5	20.00%	50.05 %	2ms
32.00KB	44	1	9.09%	50.05 %	2ms
32.00KB	44	2	9.09%	50.05 %	2ms
32.00KB	44	3	9.09%	50.05 %	2ms
32.00KB	44	4	9.09%	50.05 %	2ms
32.00KB	44	5	9.09%	50.05 %	1ms
32.00KB	48	1	0.00%	50.05 %	2ms
32.00KB	48	2	0.00%	50.05 %	2ms
32.00KB	48	3	0.00%	50.05 %	2ms
32.00KB	48	4	0.00%	50.05 %	2ms
32.00KB	48	5	0.00%	50.05 %	2ms
32.00KB	52	1	23.08%	33.40 %	1ms
32.00KB	52	2	23.08%	33.40 %	1ms
32.00KB	52	3	23.08%	33.40 %	2ms
32.00KB	52	4	23.08%	33.40 %	1ms
32.00KB	52	5	23.08%	33.40 %	1ms
32.00KB	56	1	14.29%	33.40 %	2ms
32.00KB	56	2	14.29%	33.40 %	1ms
32.00KB	56	3	14.29%	33.40 %	2ms
32.00KB	56	4	14.29%	33.40 %	1ms
32.00KB	56	5	14.29%	33.40 %	1ms
32.00KB	60	1	6.67%	33.40 %	2ms
32.00KB	60	2	6.67%	33.40 %	1ms
32.00KB	60	3	6.67%	33.40 %	1ms
32.00KB	60	4	6.67%	33.40 %	2ms
32.00KB	60	5	6.67%	33.40 %	1ms
32.00KB	64	1	0.00%	33.40 %	2ms
32.00KB	64	2	0.00%	33.40 %	1ms
32.00KB	64	3	0.00%	33.40 %	1ms
32.00KB	64	4	0.00%	33.40 %	2ms
32.00KB	64	5	0.00%	33.40 %	1ms
64.00KB	24	1	0.00%	50.01 %	12ms
64.00KB	24	2	0.00%	50.01 %	13ms
64.00KB	24	3	0.00%	50.01 %	12ms
64.00KB	24	4	0.00%	50.01 %	12ms
64.00KB	24	5	0.00%	50.01 %	12ms
64.00KB	28	1	14.29%	33.35 %	10ms
64.00KB	28	2	14.29%	33.35 %	10ms

64.00KB	28	3	14.29%	33.35 %	10ms
64.00KB	28	4	14.29%	33.35 %	10ms
64.00KB	28	5	14.29%	33.35 %	10ms
64.00KB	32	1	0.00%	33.35 %	10ms
64.00KB	32	2	0.00%	33.35 %	10ms
64.00KB	32	3	0.00%	33.35 %	10ms
64.00KB	32	4	0.00%	33.35 %	9ms
64.00KB	32	5	0.00%	33.35 %	10ms
64.00KB	36	1	33.33%	49.98 %	4ms
64.00KB	36	2	33.33%	49.98 %	5ms
64.00KB	36	3	33.33%	49.98 %	4ms
64.00KB	36	4	33.33%	49.98 %	5ms
64.00KB	36	5	33.33%	49.98 %	5ms
64.00KB	40	1	20.00%	49.98 %	4ms
64.00KB	40	2	20.00%	49.98 %	5ms
64.00KB	40	3	20.00%	49.98 %	4ms
64.00KB	40	4	20.00%	49.98 %	5ms
64.00KB	40	5	20.00%	49.98 %	4ms
64.00KB	44	1	9.09%	49.98 %	5ms
64.00KB	44	2	9.09%	49.98 %	4ms
64.00KB	44	3	9.09%	49.98 %	5ms
64.00KB	44	4	9.09%	49.98 %	4ms
64.00KB	44	5	9.09%	49.98 %	5ms
64.00KB	48	1	0.00%	49.98 %	4ms
64.00KB	48	2	0.00%	49.98 %	5ms
64.00KB	48	3	0.00%	49.98 %	4ms
64.00KB	48	4	0.00%	49.98 %	5ms
64.00KB	48	5	0.00%	49.98 %	4ms
64.00KB	52	1	23.08%	33.30 %	4ms
64.00KB	52	2	23.08%	33.30 %	3ms
64.00KB	52	3	23.08%	33.30 %	4ms
64.00KB	52	4	23.08%	33.30 %	3ms
64.00KB	52	5	23.08%	33.30 %	4ms
64.00KB	56	1	14.29%	33.30 %	3ms
64.00KB	56	2	14.29%	33.30 %	4ms
64.00KB	56	3	14.29%	33.30 %	3ms
64.00KB	56	4	14.29%	33.30 %	4ms
64.00KB	56	5	14.29%	33.30 %	3ms
64.00KB	60	1	6.67%	33.30 %	4ms
64.00KB	60	2	6.67%	33.30 %	3ms
64.00KB	60	3	6.67%	33.30 %	4ms
64.00KB	60	4	6.67%	33.30 %	3ms
64.00KB	60	5	6.67%	33.30 %	4ms
64.00KB	64	1	0.00%	33.30 %	3ms
64.00KB	64	2	0.00%	33.30 %	3ms
64.00KB	64	3	0.00%	33.30 %	4ms
64.00KB	64	4	0.00%	33.30 %	3ms
64.00KB	64	5	0.00%	33.30 %	4ms
128.00KB	24	1	0.00%	49.99 %	44ms
128.00KB	24	2	0.00%	49.99 %	44ms
128.00KB	24	3	0.00%	49.99 %	43ms
128.00KB	24	4	0.00%	49.99 %	46ms
128.00KB	24	5	0.00%	49.99 %	43ms
128.00KB	28	1	14.29%	33.33 %	35ms
128.00KB	28	2	14.29%	33.33 %	34ms
128.00KB	28	3	14.29%	33.33 %	35ms
128.00KB	28	4	14.29%	33.33 %	35ms
128.00KB	28	5	14.29%	33.33 %	35ms
128.00KB	32	1	0.00%	33.33 %	35ms
128.00KB	32	2	0.00%	33.33 %	35ms
128.00KB	32	3	0.00%	33.33 %	36ms
128.00KB	32	4	0.00%	33.33 %	35ms
128.00KB	32	5	0.00%	33.33 %	35ms
128.00KB	36	1	33.33%	50.01 %	11ms
128.00KB	36	2	33.33%	50.01 %	12ms

128.00KB	36	3	33.33%	50.01 %	13ms
128.00KB	36	4	33.33%	50.01 %	12ms
128.00KB	36	5	33.33%	50.01 %	12ms
128.00KB	40	1	20.00%	50.01 %	13ms
128.00KB	40	2	20.00%	50.01 %	12ms
128.00KB	40	3	20.00%	50.01 %	13ms
128.00KB	40	4	20.00%	50.01 %	12ms
128.00KB	40	5	20.00%	50.01 %	12ms
128.00KB	44	1	9.09%	50.01 %	13ms
128.00KB	44	2	9.09%	50.01 %	12ms
128.00KB	44	3	9.09%	50.01 %	13ms
128.00KB	44	4	9.09%	50.01 %	12ms
128.00KB	44	5	9.09%	50.01 %	12ms
128.00KB	48	1	0.00%	50.01 %	13ms
128.00KB	48	2	0.00%	50.01 %	12ms
128.00KB	48	3	0.00%	50.01 %	12ms
128.00KB	48	4	0.00%	50.01 %	13ms
128.00KB	48	5	0.00%	50.01 %	12ms
128.00KB	52	1	23.08%	33.35 %	11ms
128.00KB	52	2	23.08%	33.35 %	10ms
128.00KB	52	3	23.08%	33.35 %	10ms
128.00KB	52	4	23.08%	33.35 %	10ms
128.00KB	52	5	23.08%	33.35 %	10ms
128.00KB	56	1	14.29%	33.35 %	10ms
128.00KB	56	2	14.29%	33.35 %	10ms
128.00KB	56	3	14.29%	33.35 %	10ms
128.00KB	56	4	14.29%	33.35 %	10ms
128.00KB	56	5	14.29%	33.35 %	10ms
128.00KB	60	1	6.67%	33.35 %	10ms
128.00KB	60	2	6.67%	33.35 %	10ms
128.00KB	60	3	6.67%	33.35 %	9ms
128.00KB	60	4	6.67%	33.35 %	10ms
128.00KB	60	5	6.67%	33.35 %	10ms
128.00KB	64	1	0.00%	33.35 %	10ms
128.00KB	64	2	0.00%	33.35 %	10ms
128.00KB	64	3	0.00%	33.35 %	10ms
128.00KB	64	4	0.00%	33.35 %	10ms
128.00KB	64	5	0.00%	33.35 %	10ms
256.00KB	24	1	0.00%	50.00 %	179ms
256.00KB	24	2	0.00%	50.00 %	177ms
256.00KB	24	3	0.00%	50.00 %	177ms
256.00KB	24	4	0.00%	50.00 %	175ms
256.00KB	24	5	0.00%	50.00 %	176ms
256.00KB	28	1	14.29%	33.34 %	148ms
256.00KB	28	2	14.29%	33.34 %	148ms
256.00KB	28	3	14.29%	33.34 %	145ms
256.00KB	28	4	14.29%	33.34 %	144ms
256.00KB	28	5	14.29%	33.34 %	146ms
256.00KB	32	1	0.00%	33.34 %	146ms
256.00KB	32	2	0.00%	33.34 %	145ms
256.00KB	32	3	0.00%	33.34 %	146ms
256.00KB	32	4	0.00%	33.34 %	148ms
256.00KB	32	5	0.00%	33.34 %	147ms
256.00KB	36	1	33.33%	49.99 %	44ms
256.00KB	36	2	33.33%	49.99 %	43ms
256.00KB	36	3	33.33%	49.99 %	44ms
256.00KB	36	4	33.33%	49.99 %	43ms
256.00KB	36	5	33.33%	49.99 %	43ms
256.00KB	40	1	20.00%	49.99 %	43ms
256.00KB	40	2	20.00%	49.99 %	44ms
256.00KB	40	3	20.00%	49.99 %	43ms
256.00KB	40	4	20.00%	49.99 %	43ms
256.00KB	40	5	20.00%	49.99 %	44ms
256.00KB	44	1	9.09%	49.99 %	44ms
256.00KB	44	2	9.09%	49.99 %	43ms

256.00KB	44	3	9.09%	49.99 %	43ms
256.00KB	44	4	9.09%	49.99 %	43ms
256.00KB	44	5	9.09%	49.99 %	43ms
256.00KB	48	1	0.00%	49.99 %	44ms
256.00KB	48	2	0.00%	49.99 %	43ms
256.00KB	48	3	0.00%	49.99 %	44ms
256.00KB	48	4	0.00%	49.99 %	47ms
256.00KB	48	5	0.00%	49.99 %	43ms
256.00KB	52	1	23.08%	33.33 %	36ms
256.00KB	52	2	23.08%	33.33 %	34ms
256.00KB	52	3	23.08%	33.33 %	35ms
256.00KB	52	4	23.08%	33.33 %	35ms
256.00KB	52	5	23.08%	33.33 %	34ms
256.00KB	56	1	14.29%	33.33 %	35ms
256.00KB	56	2	14.29%	33.33 %	36ms
256.00KB	56	3	14.29%	33.33 %	36ms
256.00KB	56	4	14.29%	33.33 %	36ms
256.00KB	56	5	14.29%	33.33 %	35ms
256.00KB	60	1	6.67%	33.33 %	36ms
256.00KB	60	2	6.67%	33.33 %	36ms
256.00KB	60	3	6.67%	33.33 %	35ms
256.00KB	60	4	6.67%	33.33 %	36ms
256.00KB	60	5	6.67%	33.33 %	35ms
256.00KB	64	1	0.00%	33.33 %	35ms
256.00KB	64	2	0.00%	33.33 %	36ms
256.00KB	64	3	0.00%	33.33 %	36ms
256.00KB	64	4	0.00%	33.33 %	35ms
256.00KB	64	5	0.00%	33.33 %	35ms
512.00KB	24	1	0.00%	50.00 %	703ms
512.00KB	24	2	0.00%	50.00 %	682ms
512.00KB	24	3	0.00%	50.00 %	675ms
512.00KB	24	4	0.00%	50.00 %	683ms
512.00KB	24	5	0.00%	50.00 %	683ms
512.00KB	28	1	14.29%	33.33 %	571ms
512.00KB	28	2	14.29%	33.33 %	655ms
512.00KB	28	3	14.29%	33.33 %	588ms
512.00KB	28	4	14.29%	33.33 %	579ms
512.00KB	28	5	14.29%	33.33 %	582ms
512.00KB	32	1	0.00%	33.33 %	584ms
512.00KB	32	2	0.00%	33.33 %	586ms
512.00KB	32	3	0.00%	33.33 %	574ms
512.00KB	32	4	0.00%	33.33 %	572ms
512.00KB	32	5	0.00%	33.33 %	575ms
512.00KB	36	1	33.33%	50.00 %	174ms
512.00KB	36	2	33.33%	50.00 %	179ms
512.00KB	36	3	33.33%	50.00 %	180ms
512.00KB	36	4	33.33%	50.00 %	180ms
512.00KB	36	5	33.33%	50.00 %	177ms
512.00KB	40	1	20.00%	50.00 %	178ms
512.00KB	40	2	20.00%	50.00 %	177ms
512.00KB	40	3	20.00%	50.00 %	176ms
512.00KB	40	4	20.00%	50.00 %	177ms
512.00KB	40	5	20.00%	50.00 %	176ms
512.00KB	44	1	9.09%	50.00 %	178ms
512.00KB	44	2	9.09%	50.00 %	176ms
512.00KB	44	3	9.09%	50.00 %	176ms
512.00KB	44	4	9.09%	50.00 %	176ms
512.00KB	44	5	9.09%	50.00 %	177ms
512.00KB	48	1	0.00%	50.00 %	176ms
512.00KB	48	2	0.00%	50.00 %	180ms
512.00KB	48	3	0.00%	50.00 %	178ms
512.00KB	48	4	0.00%	50.00 %	179ms
512.00KB	48	5	0.00%	50.00 %	176ms
512.00KB	52	1	23.08%	33.34 %	146ms
512.00KB	52	2	23.08%	33.34 %	148ms

512.00KB	52	3	23.08%	33.34 %	148ms
512.00KB	52	4	23.08%	33.34 %	145ms
512.00KB	52	5	23.08%	33.34 %	146ms
512.00KB	56	1	14.29%	33.34 %	144ms
512.00KB	56	2	14.29%	33.34 %	146ms
512.00KB	56	3	14.29%	33.34 %	144ms
512.00KB	56	4	14.29%	33.34 %	145ms
512.00KB	56	5	14.29%	33.34 %	149ms
512.00KB	60	1	6.67%	33.34 %	143ms
512.00KB	60	2	6.67%	33.34 %	146ms
512.00KB	60	3	6.67%	33.34 %	146ms
512.00KB	60	4	6.67%	33.34 %	147ms
512.00KB	60	5	6.67%	33.34 %	147ms
512.00KB	64	1	0.00%	33.34 %	146ms
512.00KB	64	2	0.00%	33.34 %	150ms
512.00KB	64	3	0.00%	33.34 %	147ms
512.00KB	64	4	0.00%	33.34 %	147ms
512.00KB	64	5	0.00%	33.34 %	148ms
1024.00KB	24	1	0.00%	50.00 %	2636ms
1024.00KB	24	2	0.00%	50.00 %	2622ms
1024.00KB	24	3	0.00%	50.00 %	2631ms
1024.00KB	24	4	0.00%	50.00 %	2610ms
1024.00KB	24	5	0.00%	50.00 %	2652ms
1024.00KB	28	1	14.29%	33.33 %	2276ms
1024.00KB	28	2	14.29%	33.33 %	2312ms
1024.00KB	28	3	14.29%	33.33 %	2259ms
1024.00KB	28	4	14.29%	33.33 %	2261ms
1024.00KB	28	5	14.29%	33.33 %	2257ms
1024.00KB	32	1	0.00%	33.33 %	2275ms
1024.00KB	32	2	0.00%	33.33 %	2300ms
1024.00KB	32	3	0.00%	33.33 %	2315ms
1024.00KB	32	4	0.00%	33.33 %	2311ms
1024.00KB	32	5	0.00%	33.33 %	2306ms
1024.00KB	36	1	33.33%	50.00 %	691ms
1024.00KB	36	2	33.33%	50.00 %	687ms
1024.00KB	36	3	33.33%	50.00 %	692ms
1024.00KB	36	4	33.33%	50.00 %	680ms
1024.00KB	36	5	33.33%	50.00 %	683ms
1024.00KB	40	1	20.00%	50.00 %	677ms
1024.00KB	40	2	20.00%	50.00 %	678ms
1024.00KB	40	3	20.00%	50.00 %	674ms
1024.00KB	40	4	20.00%	50.00 %	683ms
1024.00KB	40	5	20.00%	50.00 %	681ms
1024.00KB	44	1	9.09%	50.00 %	676ms
1024.00KB	44	2	9.09%	50.00 %	671ms
1024.00KB	44	3	9.09%	50.00 %	685ms
1024.00KB	44	4	9.09%	50.00 %	684ms
1024.00KB	44	5	9.09%	50.00 %	680ms
1024.00KB	48	1	0.00%	50.00 %	674ms
1024.00KB	48	2	0.00%	50.00 %	681ms
1024.00KB	48	3	0.00%	50.00 %	680ms
1024.00KB	48	4	0.00%	50.00 %	737ms
1024.00KB	48	5	0.00%	50.00 %	689ms
1024.00KB	52	1	23.08%	33.33 %	587ms
1024.00KB	52	2	23.08%	33.33 %	581ms
1024.00KB	52	3	23.08%	33.33 %	586ms
1024.00KB	52	4	23.08%	33.33 %	594ms
1024.00KB	52	5	23.08%	33.33 %	581ms
1024.00KB	56	1	14.29%	33.33 %	577ms
1024.00KB	56	2	14.29%	33.33 %	583ms
1024.00KB	56	3	14.29%	33.33 %	575ms
1024.00KB	56	4	14.29%	33.33 %	576ms
1024.00KB	56	5	14.29%	33.33 %	586ms
1024.00KB	60	1	6.67%	33.33 %	591ms
1024.00KB	60	2	6.67%	33.33 %	592ms

1024.00KB	60	3	6.67%	33.33 %	583ms
1024.00KB	60	4	6.67%	33.33 %	588ms
1024.00KB	60	5	6.67%	33.33 %	589ms
1024.00KB	64	1	0.00%	33.33 %	583ms
1024.00KB	64	2	0.00%	33.33 %	585ms
1024.00KB	64	3	0.00%	33.33 %	581ms
1024.00KB	64	4	0.00%	33.33 %	589ms
1024.00KB	64	5	0.00%	33.33 %	581ms
2.00MB	48	1	0.00%	50.00 %	2660ms
2.00MB	48	2	0.00%	50.00 %	2638ms
2.00MB	48	3	0.00%	50.00 %	2688ms
2.00MB	48	4	0.00%	50.00 %	2650ms
2.00MB	48	5	0.00%	50.00 %	2672ms
2.00MB	56	1	14.29%	33.33 %	2299ms
2.00MB	56	2	14.29%	33.33 %	2258ms
2.00MB	56	3	14.29%	33.33 %	2371ms
2.00MB	56	4	14.29%	33.33 %	2319ms
2.00MB	56	5	14.29%	33.33 %	2338ms
2.00MB	64	1	0.00%	33.33 %	2280ms
2.00MB	64	2	0.00%	33.33 %	2294ms
2.00MB	64	3	0.00%	33.33 %	2282ms
2.00MB	64	4	0.00%	33.33 %	2266ms
2.00MB	64	5	0.00%	33.33 %	2302ms
2.00MB	72	1	33.33%	50.00 %	673ms
2.00MB	72	2	33.33%	50.00 %	675ms
2.00MB	72	3	33.33%	50.00 %	679ms
2.00MB	72	4	33.33%	50.00 %	679ms
2.00MB	72	5	33.33%	50.00 %	682ms
2.00MB	80	1	20.00%	50.00 %	692ms
2.00MB	80	2	20.00%	50.00 %	686ms
2.00MB	80	3	20.00%	50.00 %	683ms
2.00MB	80	4	20.00%	50.00 %	695ms
2.00MB	80	5	20.00%	50.00 %	686ms
2.00MB	88	1	9.09%	50.00 %	686ms
2.00MB	88	2	9.09%	50.00 %	673ms
2.00MB	88	3	9.09%	50.00 %	680ms
2.00MB	88	4	9.09%	50.00 %	682ms
2.00MB	88	5	9.09%	50.00 %	690ms
2.00MB	96	1	0.00%	50.00 %	747ms
2.00MB	96	2	0.00%	50.00 %	683ms
2.00MB	96	3	0.00%	50.00 %	682ms
2.00MB	96	4	0.00%	50.00 %	685ms
2.00MB	96	5	0.00%	50.00 %	702ms
2.00MB	104	1	23.08%	33.33 %	585ms
2.00MB	104	2	23.08%	33.33 %	586ms
2.00MB	104	3	23.08%	33.33 %	584ms
2.00MB	104	4	23.08%	33.33 %	589ms
2.00MB	104	5	23.08%	33.33 %	583ms
2.00MB	112	1	14.29%	33.33 %	583ms
2.00MB	112	2	14.29%	33.33 %	586ms
2.00MB	112	3	14.29%	33.33 %	581ms
2.00MB	112	4	14.29%	33.33 %	584ms
2.00MB	112	5	14.29%	33.33 %	589ms
2.00MB	120	1	6.67%	33.33 %	585ms
2.00MB	120	2	6.67%	33.33 %	590ms
2.00MB	120	3	6.67%	33.33 %	572ms
2.00MB	120	4	6.67%	33.33 %	579ms
2.00MB	120	5	6.67%	33.33 %	579ms
2.00MB	128	1	0.00%	33.33 %	578ms
2.00MB	128	2	0.00%	33.33 %	587ms
2.00MB	128	3	0.00%	33.33 %	586ms
2.00MB	128	4	0.00%	33.33 %	584ms
2.00MB	128	5	0.00%	33.33 %	584ms
4.00MB	96	1	0.00%	50.00 %	2654ms
4.00MB	96	2	0.00%	50.00 %	2646ms

4.00MB	96	3	0.00%	50.00 %	2663ms
4.00MB	96	4	0.00%	50.00 %	2639ms
4.00MB	96	5	0.00%	50.00 %	2653ms
4.00MB	112	1	14.29%	33.33 %	2258ms
4.00MB	112	2	14.29%	33.33 %	2281ms
4.00MB	112	3	14.29%	33.33 %	2282ms
4.00MB	112	4	14.29%	33.33 %	2275ms
4.00MB	112	5	14.29%	33.33 %	2261ms
4.00MB	128	1	0.00%	33.33 %	2264ms
4.00MB	128	2	0.00%	33.33 %	2238ms
4.00MB	128	3	0.00%	33.33 %	2281ms
4.00MB	128	4	0.00%	33.33 %	2269ms
4.00MB	128	5	0.00%	33.33 %	2286ms
4.00MB	144	1	33.33%	50.00 %	675ms
4.00MB	144	2	33.33%	50.00 %	674ms
4.00MB	144	3	33.33%	50.00 %	680ms
4.00MB	144	4	33.33%	50.00 %	688ms
4.00MB	144	5	33.33%	50.00 %	690ms
4.00MB	160	1	20.00%	50.00 %	686ms
4.00MB	160	2	20.00%	50.00 %	680ms
4.00MB	160	3	20.00%	50.00 %	673ms
4.00MB	160	4	20.00%	50.00 %	677ms
4.00MB	160	5	20.00%	50.00 %	687ms
4.00MB	176	1	9.09%	50.00 %	691ms
4.00MB	176	2	9.09%	50.00 %	687ms
4.00MB	176	3	9.09%	50.00 %	688ms
4.00MB	176	4	9.09%	50.00 %	870ms
4.00MB	176	5	9.09%	50.00 %	722ms
4.00MB	192	1	0.00%	50.00 %	710ms
4.00MB	192	2	0.00%	50.00 %	691ms
4.00MB	192	3	0.00%	50.00 %	684ms
4.00MB	192	4	0.00%	50.00 %	706ms
4.00MB	192	5	0.00%	50.00 %	715ms
4.00MB	208	1	23.08%	33.33 %	599ms
4.00MB	208	2	23.08%	33.33 %	586ms
4.00MB	208	3	23.08%	33.33 %	586ms
4.00MB	208	4	23.08%	33.33 %	588ms
4.00MB	208	5	23.08%	33.33 %	586ms
4.00MB	224	1	14.29%	33.33 %	583ms
4.00MB	224	2	14.29%	33.33 %	577ms
4.00MB	224	3	14.29%	33.33 %	583ms
4.00MB	224	4	14.29%	33.33 %	581ms
4.00MB	224	5	14.29%	33.33 %	581ms
4.00MB	240	1	6.67%	33.33 %	585ms
4.00MB	240	2	6.67%	33.33 %	582ms
4.00MB	240	3	6.67%	33.33 %	592ms
4.00MB	240	4	6.67%	33.33 %	578ms
4.00MB	240	5	6.67%	33.33 %	591ms
4.00MB	256	1	0.00%	33.33 %	593ms
4.00MB	256	2	0.00%	33.33 %	576ms
4.00MB	256	3	0.00%	33.33 %	584ms
4.00MB	256	4	0.00%	33.33 %	588ms
4.00MB	256	5	0.00%	33.33 %	584ms
8.00MB	192	1	0.00%	50.00 %	2666ms
8.00MB	192	2	0.00%	50.00 %	2717ms
8.00MB	192	3	0.00%	50.00 %	2662ms
8.00MB	192	4	0.00%	50.00 %	2671ms
8.00MB	192	5	0.00%	50.00 %	2714ms
8.00MB	224	1	14.29%	33.33 %	2283ms
8.00MB	224	2	14.29%	33.33 %	2281ms
8.00MB	224	3	14.29%	33.33 %	2290ms
8.00MB	224	4	14.29%	33.33 %	2306ms
8.00MB	224	5	14.29%	33.33 %	2298ms
8.00MB	256	1	0.00%	33.33 %	2289ms
8.00MB	256	2	0.00%	33.33 %	2280ms

8.00MB	256	3	0.00%	33.33 %	2291ms
8.00MB	256	4	0.00%	33.33 %	2296ms
8.00MB	256	5	0.00%	33.33 %	2309ms
8.00MB	288	1	33.33%	50.00 %	686ms
8.00MB	288	2	33.33%	50.00 %	681ms
8.00MB	288	3	33.33%	50.00 %	680ms
8.00MB	288	4	33.33%	50.00 %	697ms
8.00MB	288	5	33.33%	50.00 %	694ms
8.00MB	320	1	20.00%	50.00 %	695ms
8.00MB	320	2	20.00%	50.00 %	689ms
8.00MB	320	3	20.00%	50.00 %	692ms
8.00MB	320	4	20.00%	50.00 %	688ms
8.00MB	320	5	20.00%	50.00 %	686ms
8.00MB	352	1	9.09%	50.00 %	769ms
8.00MB	352	2	9.09%	50.00 %	693ms
8.00MB	352	3	9.09%	50.00 %	679ms
8.00MB	352	4	9.09%	50.00 %	687ms
8.00MB	352	5	9.09%	50.00 %	688ms
8.00MB	384	1	0.00%	50.00 %	683ms
8.00MB	384	2	0.00%	50.00 %	685ms
8.00MB	384	3	0.00%	50.00 %	684ms
8.00MB	384	4	0.00%	50.00 %	702ms
8.00MB	384	5	0.00%	50.00 %	697ms
8.00MB	416	1	23.08%	33.33 %	594ms
8.00MB	416	2	23.08%	33.33 %	581ms
8.00MB	416	3	23.08%	33.33 %	585ms
8.00MB	416	4	23.08%	33.33 %	586ms
8.00MB	416	5	23.08%	33.33 %	587ms
8.00MB	448	1	14.29%	33.33 %	591ms
8.00MB	448	2	14.29%	33.33 %	587ms
8.00MB	448	3	14.29%	33.33 %	586ms
8.00MB	448	4	14.29%	33.33 %	588ms
8.00MB	448	5	14.29%	33.33 %	590ms
8.00MB	480	1	6.67%	33.33 %	584ms
8.00MB	480	2	6.67%	33.33 %	587ms
8.00MB	480	3	6.67%	33.33 %	589ms
8.00MB	480	4	6.67%	33.33 %	590ms
8.00MB	480	5	6.67%	33.33 %	586ms
8.00MB	512	1	0.00%	33.33 %	592ms
8.00MB	512	2	0.00%	33.33 %	584ms
8.00MB	512	3	0.00%	33.33 %	580ms
8.00MB	512	4	0.00%	33.33 %	580ms
8.00MB	512	5	0.00%	33.33 %	585ms
16.00MB	384	1	0.00%	50.00 %	2659ms
16.00MB	384	2	0.00%	50.00 %	2624ms
16.00MB	384	3	0.00%	50.00 %	2647ms
16.00MB	384	4	0.00%	50.00 %	2682ms
16.00MB	384	5	0.00%	50.00 %	2695ms
16.00MB	448	1	14.29%	33.33 %	2286ms
16.00MB	448	2	14.29%	33.33 %	2315ms
16.00MB	448	3	14.29%	33.33 %	2309ms
16.00MB	448	4	14.29%	33.33 %	2298ms
16.00MB	448	5	14.29%	33.33 %	2296ms
16.00MB	512	1	0.00%	33.33 %	2296ms
16.00MB	512	2	0.00%	33.33 %	2297ms
16.00MB	512	3	0.00%	33.33 %	2278ms
16.00MB	512	4	0.00%	33.33 %	2319ms
16.00MB	512	5	0.00%	33.33 %	2297ms
16.00MB	576	1	33.33%	50.00 %	691ms
16.00MB	576	2	33.33%	50.00 %	691ms
16.00MB	576	3	33.33%	50.00 %	719ms
16.00MB	576	4	33.33%	50.00 %	699ms
16.00MB	576	5	33.33%	50.00 %	695ms
16.00MB	640	1	20.00%	50.00 %	693ms
16.00MB	640	2	20.00%	50.00 %	696ms

16.00MB	640	3	20.00%	50.00 %	760ms
16.00MB	640	4	20.00%	50.00 %	715ms
16.00MB	640	5	20.00%	50.00 %	693ms
16.00MB	704	1	9.09%	50.00 %	692ms
16.00MB	704	2	9.09%	50.00 %	697ms
16.00MB	704	3	9.09%	50.00 %	697ms
16.00MB	704	4	9.09%	50.00 %	697ms
16.00MB	704	5	9.09%	50.00 %	688ms
16.00MB	768	1	0.00%	50.00 %	691ms
16.00MB	768	2	0.00%	50.00 %	693ms
16.00MB	768	3	0.00%	50.00 %	694ms
16.00MB	768	4	0.00%	50.00 %	687ms
16.00MB	768	5	0.00%	50.00 %	681ms
16.00MB	832	1	23.08%	33.33 %	589ms
16.00MB	832	2	23.08%	33.33 %	585ms
16.00MB	832	3	23.08%	33.33 %	586ms
16.00MB	832	4	23.08%	33.33 %	592ms
16.00MB	832	5	23.08%	33.33 %	591ms
16.00MB	896	1	14.29%	33.33 %	594ms
16.00MB	896	2	14.29%	33.33 %	580ms
16.00MB	896	3	14.29%	33.33 %	586ms
16.00MB	896	4	14.29%	33.33 %	588ms
16.00MB	896	5	14.29%	33.33 %	592ms
16.00MB	960	1	6.67%	33.33 %	586ms
16.00MB	960	2	6.67%	33.33 %	597ms
16.00MB	960	3	6.67%	33.33 %	588ms
16.00MB	960	4	6.67%	33.33 %	585ms
16.00MB	960	5	6.67%	33.33 %	583ms
16.00MB	1024	1	0.00%	33.33 %	582ms
16.00MB	1024	2	0.00%	33.33 %	582ms
16.00MB	1024	3	0.00%	33.33 %	590ms
16.00MB	1024	4	0.00%	33.33 %	587ms
16.00MB	1024	5	0.00%	33.33 %	582ms
32.00MB	768	1	0.00%	50.00 %	2672ms
32.00MB	768	2	0.00%	50.00 %	2671ms
32.00MB	768	3	0.00%	50.00 %	2673ms
32.00MB	768	4	0.00%	50.00 %	2671ms
32.00MB	768	5	0.00%	50.00 %	2666ms
32.00MB	896	1	14.29%	33.33 %	2293ms
32.00MB	896	2	14.29%	33.33 %	2280ms
32.00MB	896	3	14.29%	33.33 %	2297ms
32.00MB	896	4	14.29%	33.33 %	2275ms
32.00MB	896	5	14.29%	33.33 %	2295ms
32.00MB	1024	1	0.00%	33.33 %	2303ms
32.00MB	1024	2	0.00%	33.33 %	2256ms
32.00MB	1024	3	0.00%	33.33 %	2295ms
32.00MB	1024	4	0.00%	33.33 %	2286ms
32.00MB	1024	5	0.00%	33.33 %	2264ms
32.00MB	1152	1	33.33%	50.00 %	727ms
32.00MB	1152	2	33.33%	50.00 %	692ms
32.00MB	1152	3	33.33%	50.00 %	692ms
32.00MB	1152	4	33.33%	50.00 %	692ms
32.00MB	1152	5	33.33%	50.00 %	713ms
32.00MB	1280	1	20.00%	50.00 %	731ms
32.00MB	1280	2	20.00%	50.00 %	698ms
32.00MB	1280	3	20.00%	50.00 %	699ms
32.00MB	1280	4	20.00%	50.00 %	699ms
32.00MB	1280	5	20.00%	50.00 %	694ms
32.00MB	1408	1	9.09%	50.00 %	690ms
32.00MB	1408	2	9.09%	50.00 %	692ms
32.00MB	1408	3	9.09%	50.00 %	696ms
32.00MB	1408	4	9.09%	50.00 %	697ms
32.00MB	1408	5	9.09%	50.00 %	684ms
32.00MB	1536	1	0.00%	50.00 %	698ms
32.00MB	1536	2	0.00%	50.00 %	696ms

32.00MB	1536	3	0.00%	50.00 %	698ms
32.00MB	1536	4	0.00%	50.00 %	696ms
32.00MB	1536	5	0.00%	50.00 %	696ms
32.00MB	1664	1	23.08%	33.33 %	596ms
32.00MB	1664	2	23.08%	33.33 %	588ms
32.00MB	1664	3	23.08%	33.33 %	590ms
32.00MB	1664	4	23.08%	33.33 %	588ms
32.00MB	1664	5	23.08%	33.33 %	591ms
32.00MB	1792	1	14.29%	33.33 %	586ms
32.00MB	1792	2	14.29%	33.33 %	598ms
32.00MB	1792	3	14.29%	33.33 %	587ms
32.00MB	1792	4	14.29%	33.33 %	599ms
32.00MB	1792	5	14.29%	33.33 %	591ms
32.00MB	1920	1	6.67%	33.33 %	586ms
32.00MB	1920	2	6.67%	33.33 %	582ms
32.00MB	1920	3	6.67%	33.33 %	589ms
32.00MB	1920	4	6.67%	33.33 %	599ms
32.00MB	1920	5	6.67%	33.33 %	594ms
32.00MB	2048	1	0.00%	33.33 %	588ms
32.00MB	2048	2	0.00%	33.33 %	595ms
32.00MB	2048	3	0.00%	33.33 %	592ms
32.00MB	2048	4	0.00%	33.33 %	587ms
32.00MB	2048	5	0.00%	33.33 %	593ms
64.00MB	1536	1	0.00%	50.00 %	2677ms
64.00MB	1536	2	0.00%	50.00 %	2683ms
64.00MB	1536	3	0.00%	50.00 %	2649ms
64.00MB	1536	4	0.00%	50.00 %	2672ms
64.00MB	1536	5	0.00%	50.00 %	2657ms
64.00MB	1792	1	14.29%	33.33 %	2270ms
64.00MB	1792	2	14.29%	33.33 %	2305ms
64.00MB	1792	3	14.29%	33.33 %	2318ms
64.00MB	1792	4	14.29%	33.33 %	2314ms
64.00MB	1792	5	14.29%	33.33 %	2312ms
64.00MB	2048	1	0.00%	33.33 %	2300ms
64.00MB	2048	2	0.00%	33.33 %	2280ms
64.00MB	2048	3	0.00%	33.33 %	2277ms
64.00MB	2048	4	0.00%	33.33 %	2287ms
64.00MB	2048	5	0.00%	33.33 %	2309ms
64.00MB	2304	1	33.33%	50.00 %	701ms
64.00MB	2304	2	33.33%	50.00 %	741ms
64.00MB	2304	3	33.33%	50.00 %	718ms
64.00MB	2304	4	33.33%	50.00 %	702ms
64.00MB	2304	5	33.33%	50.00 %	701ms
64.00MB	2560	1	20.00%	50.00 %	690ms
64.00MB	2560	2	20.00%	50.00 %	694ms
64.00MB	2560	3	20.00%	50.00 %	697ms
64.00MB	2560	4	20.00%	50.00 %	699ms
64.00MB	2560	5	20.00%	50.00 %	703ms
64.00MB	2816	1	9.09%	50.00 %	701ms
64.00MB	2816	2	9.09%	50.00 %	695ms
64.00MB	2816	3	9.09%	50.00 %	702ms
64.00MB	2816	4	9.09%	50.00 %	700ms
64.00MB	2816	5	9.09%	50.00 %	701ms
64.00MB	3072	1	0.00%	50.00 %	689ms
64.00MB	3072	2	0.00%	50.00 %	695ms
64.00MB	3072	3	0.00%	50.00 %	701ms
64.00MB	3072	4	0.00%	50.00 %	701ms
64.00MB	3072	5	0.00%	50.00 %	696ms
64.00MB	3328	1	23.08%	33.33 %	597ms
64.00MB	3328	2	23.08%	33.33 %	602ms
64.00MB	3328	3	23.08%	33.33 %	593ms
64.00MB	3328	4	23.08%	33.33 %	593ms
64.00MB	3328	5	23.08%	33.33 %	604ms
64.00MB	3584	1	14.29%	33.33 %	594ms
64.00MB	3584	2	14.29%	33.33 %	590ms

64.00MB	3584	3	14.29%	33.33 %	597ms
64.00MB	3584	4	14.29%	33.33 %	599ms
64.00MB	3584	5	14.29%	33.33 %	606ms
64.00MB	3840	1	6.67%	33.33 %	602ms
64.00MB	3840	2	6.67%	33.33 %	599ms
64.00MB	3840	3	6.67%	33.33 %	594ms
64.00MB	3840	4	6.67%	33.33 %	589ms
64.00MB	3840	5	6.67%	33.33 %	603ms
64.00MB	4096	1	0.00%	33.33 %	606ms
64.00MB	4096	2	0.00%	33.33 %	597ms
64.00MB	4096	3	0.00%	33.33 %	599ms
64.00MB	4096	4	0.00%	33.33 %	593ms
64.00MB	4096	5	0.00%	33.33 %	605ms
128.00MB	3072	1	0.00%	50.00 %	2700ms
128.00MB	3072	2	0.00%	50.00 %	2712ms
128.00MB	3072	3	0.00%	50.00 %	2680ms
128.00MB	3072	4	0.00%	50.00 %	2673ms
128.00MB	3072	5	0.00%	50.00 %	2668ms
128.00MB	3584	1	14.29%	33.33 %	2323ms
128.00MB	3584	2	14.29%	33.33 %	2300ms
128.00MB	3584	3	14.29%	33.33 %	2307ms
128.00MB	3584	4	14.29%	33.33 %	2322ms
128.00MB	3584	5	14.29%	33.33 %	2294ms
128.00MB	4096	1	0.00%	33.33 %	2311ms
128.00MB	4096	2	0.00%	33.33 %	2345ms
128.00MB	4096	3	0.00%	33.33 %	2300ms
128.00MB	4096	4	0.00%	33.33 %	2305ms
128.00MB	4096	5	0.00%	33.33 %	2452ms
128.00MB	4608	1	33.33%	50.00 %	702ms
128.00MB	4608	2	33.33%	50.00 %	700ms
128.00MB	4608	3	33.33%	50.00 %	697ms
128.00MB	4608	4	33.33%	50.00 %	705ms
128.00MB	4608	5	33.33%	50.00 %	713ms
128.00MB	5120	1	20.00%	50.00 %	704ms
128.00MB	5120	2	20.00%	50.00 %	707ms
128.00MB	5120	3	20.00%	50.00 %	704ms
128.00MB	5120	4	20.00%	50.00 %	706ms
128.00MB	5120	5	20.00%	50.00 %	704ms
128.00MB	5632	1	9.09%	50.00 %	712ms
128.00MB	5632	2	9.09%	50.00 %	705ms
128.00MB	5632	3	9.09%	50.00 %	708ms
128.00MB	5632	4	9.09%	50.00 %	704ms
128.00MB	5632	5	9.09%	50.00 %	708ms
128.00MB	6144	1	0.00%	50.00 %	701ms
128.00MB	6144	2	0.00%	50.00 %	703ms
128.00MB	6144	3	0.00%	50.00 %	704ms
128.00MB	6144	4	0.00%	50.00 %	705ms
128.00MB	6144	5	0.00%	50.00 %	707ms
128.00MB	6656	1	23.08%	33.33 %	611ms
128.00MB	6656	2	23.08%	33.33 %	605ms
128.00MB	6656	3	23.08%	33.33 %	600ms
128.00MB	6656	4	23.08%	33.33 %	605ms
128.00MB	6656	5	23.08%	33.33 %	596ms
128.00MB	7168	1	14.29%	33.33 %	596ms
128.00MB	7168	2	14.29%	33.33 %	598ms
128.00MB	7168	3	14.29%	33.33 %	604ms
128.00MB	7168	4	14.29%	33.33 %	602ms
128.00MB	7168	5	14.29%	33.33 %	602ms
128.00MB	7680	1	6.67%	33.33 %	601ms
128.00MB	7680	2	6.67%	33.33 %	607ms
128.00MB	7680	3	6.67%	33.33 %	600ms
128.00MB	7680	4	6.67%	33.33 %	602ms
128.00MB	7680	5	6.67%	33.33 %	603ms
128.00MB	8192	1	0.00%	33.33 %	600ms
128.00MB	8192	2	0.00%	33.33 %	594ms

128.00MB	8192	3	0.00%	33.33 %	593ms
128.00MB	8192	4	0.00%	33.33 %	596ms
128.00MB	8192	5	0.00%	33.33 %	598ms
256.00MB	6144	1	0.00%	50.00 %	2672ms
256.00MB	6144	2	0.00%	50.00 %	2669ms
256.00MB	6144	3	0.00%	50.00 %	2657ms
256.00MB	6144	4	0.00%	50.00 %	2682ms
256.00MB	6144	5	0.00%	50.00 %	2710ms
256.00MB	7168	1	14.29%	33.33 %	2307ms
256.00MB	7168	2	14.29%	33.33 %	2318ms
256.00MB	7168	3	14.29%	33.33 %	2305ms
256.00MB	7168	4	14.29%	33.33 %	2297ms
256.00MB	7168	5	14.29%	33.33 %	2303ms
256.00MB	8192	1	0.00%	33.33 %	2318ms
256.00MB	8192	2	0.00%	33.33 %	2336ms
256.00MB	8192	3	0.00%	33.33 %	2323ms
256.00MB	8192	4	0.00%	33.33 %	2447ms
256.00MB	8192	5	0.00%	33.33 %	2313ms
256.00MB	9216	1	33.33%	50.00 %	707ms
256.00MB	9216	2	33.33%	50.00 %	712ms
256.00MB	9216	3	33.33%	50.00 %	710ms
256.00MB	9216	4	33.33%	50.00 %	718ms
256.00MB	9216	5	33.33%	50.00 %	713ms
256.00MB	10240	1	20.00%	50.00 %	714ms
256.00MB	10240	2	20.00%	50.00 %	713ms
256.00MB	10240	3	20.00%	50.00 %	703ms
256.00MB	10240	4	20.00%	50.00 %	706ms
256.00MB	10240	5	20.00%	50.00 %	711ms
256.00MB	11264	1	9.09%	50.00 %	699ms
256.00MB	11264	2	9.09%	50.00 %	699ms
256.00MB	11264	3	9.09%	50.00 %	698ms
256.00MB	11264	4	9.09%	50.00 %	702ms
256.00MB	11264	5	9.09%	50.00 %	706ms
256.00MB	12288	1	0.00%	50.00 %	704ms
256.00MB	12288	2	0.00%	50.00 %	713ms
256.00MB	12288	3	0.00%	50.00 %	709ms
256.00MB	12288	4	0.00%	50.00 %	712ms
256.00MB	12288	5	0.00%	50.00 %	697ms
256.00MB	13312	1	23.08%	33.33 %	599ms
256.00MB	13312	2	23.08%	33.33 %	598ms
256.00MB	13312	3	23.08%	33.33 %	594ms
256.00MB	13312	4	23.08%	33.33 %	601ms
256.00MB	13312	5	23.08%	33.33 %	593ms
256.00MB	14336	1	14.29%	33.33 %	599ms
256.00MB	14336	2	14.29%	33.33 %	602ms
256.00MB	14336	3	14.29%	33.33 %	600ms
256.00MB	14336	4	14.29%	33.33 %	602ms
256.00MB	14336	5	14.29%	33.33 %	594ms
256.00MB	15360	1	6.67%	33.33 %	597ms
256.00MB	15360	2	6.67%	33.33 %	596ms
256.00MB	15360	3	6.67%	33.33 %	596ms
256.00MB	15360	4	6.67%	33.33 %	599ms
256.00MB	15360	5	6.67%	33.33 %	597ms
256.00MB	16384	1	0.00%	33.33 %	602ms
256.00MB	16384	2	0.00%	33.33 %	607ms
256.00MB	16384	3	0.00%	33.33 %	613ms
256.00MB	16384	4	0.00%	33.33 %	604ms
256.00MB	16384	5	0.00%	33.33 %	600ms

2.2. Test for oneBin O(n)

The Experiment Statistics are as follows
The Average Internal Fragmentation was 4.90%
The Average External Fragmentation was 8.79%
The Average Time Taken in ms. per MB was 1.49

```
------------------------------------------------------------------------
------------------------------------------------------------------------
           Starting the experiment for one bin
------------------------------------------------------------------------
------------------------------------------------------------------------
   MemSize || Bin|| Cnt|| IntFrag|| ExtFrag    || TimeTaken
      32B  || 28 || 1  || 14.29% ||  100.00 %  || 0ms
      32B  || 28 || 2  || 14.29% ||  100.00 %  || 0ms
      32B  || 28 || 3  || 14.29% ||  100.00 %  || 0ms
      32B  || 28 || 4  || 14.29% ||  100.00 %  || 0ms
      32B  || 28 || 5  || 14.29% ||  100.00 %  || 0ms
      32B  || 32 || 1  ||  0.00% ||  100.00 %  || 0ms
      32B  || 32 || 2  ||  0.00% ||  100.00 %  || 0ms
      32B  || 32 || 3  ||  0.00% ||  100.00 %  || 0ms
      32B  || 32 || 4  ||  0.00% ||  100.00 %  || 0ms
      32B  || 32 || 5  ||  0.00% ||  100.00 %  || 0ms
      64B  || 28 || 1  || 14.29% ||  50.00 %   || 0ms
      64B  || 28 || 2  || 14.29% ||  50.00 %   || 0ms
      64B  || 28 || 3  || 14.29% ||  50.00 %   || 0ms
      64B  || 28 || 4  || 14.29% ||  50.00 %   || 0ms
      64B  || 28 || 5  || 14.29% ||  50.00 %   || 0ms
      64B  || 32 || 1  ||  0.00% ||  50.00 %   || 0ms
      64B  || 32 || 2  ||  0.00% ||  50.00 %   || 0ms
      64B  || 32 || 3  ||  0.00% ||  50.00 %   || 0ms
      64B  || 32 || 4  ||  0.00% ||  50.00 %   || 0ms
      64B  || 32 || 5  ||  0.00% ||  50.00 %   || 0ms
      64B  || 36 || 1  || 11.11% ||  100.00 %  || 0ms
      64B  || 36 || 2  || 11.11% ||  100.00 %  || 0ms
      64B  || 36 || 3  || 11.11% ||  100.00 %  || 0ms
      64B  || 36 || 4  || 11.11% ||  100.00 %  || 0ms
      64B  || 36 || 5  || 11.11% ||  100.00 %  || 0ms
      64B  || 40 || 1  ||  0.00% ||  100.00 %  || 0ms
      64B  || 40 || 2  ||  0.00% ||  100.00 %  || 0ms
      64B  || 40 || 3  ||  0.00% ||  100.00 %  || 0ms
      64B  || 40 || 4  ||  0.00% ||  100.00 %  || 0ms
      64B  || 40 || 5  ||  0.00% ||  100.00 %  || 0ms
      64B  || 44 || 1  ||  9.09% ||  100.00 %  || 0ms
      64B  || 44 || 2  ||  9.09% ||  100.00 %  || 0ms
      64B  || 44 || 3  ||  9.09% ||  100.00 %  || 0ms
      64B  || 44 || 4  ||  9.09% ||  100.00 %  || 0ms
      64B  || 44 || 5  ||  9.09% ||  100.00 %  || 0ms
      64B  || 48 || 1  ||  0.00% ||  100.00 %  || 0ms
      64B  || 48 || 2  ||  0.00% ||  100.00 %  || 0ms
      64B  || 48 || 3  ||  0.00% ||  100.00 %  || 0ms
      64B  || 48 || 4  ||  0.00% ||  100.00 %  || 0ms
      64B  || 48 || 5  ||  0.00% ||  100.00 %  || 0ms
      64B  || 52 || 1  ||  7.69% ||  100.00 %  || 0ms
      64B  || 52 || 2  ||  7.69% ||  100.00 %  || 0ms
      64B  || 52 || 3  ||  7.69% ||  100.00 %  || 0ms
      64B  || 52 || 4  ||  7.69% ||  100.00 %  || 0ms
      64B  || 52 || 5  ||  7.69% ||  100.00 %  || 0ms
      64B  || 56 || 1  ||  0.00% ||  100.00 %  || 0ms
      64B  || 56 || 2  ||  0.00% ||  100.00 %  || 0ms
      64B  || 56 || 3  ||  0.00% ||  100.00 %  || 0ms
      64B  || 56 || 4  ||  0.00% ||  100.00 %  || 0ms
      64B  || 56 || 5  ||  0.00% ||  100.00 %  || 0ms
      64B  || 60 || 1  ||  6.67% ||  100.00 %  || 0ms
```

```
   64B  || 60 || 2  ||  6.67% ||  100.00 %  || 0ms
   64B  || 60 || 3  ||  6.67% ||  100.00 %  || 0ms
   64B  || 60 || 4  ||  6.67% ||  100.00 %  || 0ms
   64B  || 60 || 5  ||  6.67% ||  100.00 %  || 0ms
   64B  || 64 || 1  ||  0.00% ||  100.00 %  || 0ms
   64B  || 64 || 2  ||  0.00% ||  100.00 %  || 0ms
   64B  || 64 || 3  ||  0.00% ||  100.00 %  || 0ms
   64B  || 64 || 4  ||  0.00% ||  100.00 %  || 0ms
   64B  || 64 || 5  ||  0.00% ||  100.00 %  || 0ms
  128B  || 28 || 1  || 14.29% ||  25.00 %   || 0ms
  128B  || 28 || 2  || 14.29% ||  25.00 %   || 0ms
  128B  || 28 || 3  || 14.29% ||  25.00 %   || 0ms
  128B  || 28 || 4  || 14.29% ||  25.00 %   || 0ms
  128B  || 28 || 5  || 14.29% ||  25.00 %   || 0ms
  128B  || 32 || 1  ||  0.00% ||  25.00 %   || 0ms
  128B  || 32 || 2  ||  0.00% ||  25.00 %   || 0ms
  128B  || 32 || 3  ||  0.00% ||  25.00 %   || 0ms
  128B  || 32 || 4  ||  0.00% ||  25.00 %   || 0ms
  128B  || 32 || 5  ||  0.00% ||  25.00 %   || 0ms
  128B  || 36 || 1  || 11.11% ||  37.50 %   || 0ms
  128B  || 36 || 2  || 11.11% ||  37.50 %   || 0ms
  128B  || 36 || 3  || 11.11% ||  37.50 %   || 0ms
  128B  || 36 || 4  || 11.11% ||  37.50 %   || 0ms
  128B  || 36 || 5  || 11.11% ||  37.50 %   || 0ms
  128B  || 40 || 1  ||  0.00% ||  37.50 %   || 0ms
  128B  || 40 || 2  ||  0.00% ||  37.50 %   || 0ms
  128B  || 40 || 3  ||  0.00% ||  37.50 %   || 0ms
  128B  || 40 || 4  ||  0.00% ||  37.50 %   || 0ms
  128B  || 40 || 5  ||  0.00% ||  37.50 %   || 0ms
  128B  || 44 || 1  ||  9.09% ||  62.50 %   || 0ms
  128B  || 44 || 2  ||  9.09% ||  62.50 %   || 0ms
  128B  || 44 || 3  ||  9.09% ||  62.50 %   || 0ms
  128B  || 44 || 4  ||  9.09% ||  62.50 %   || 0ms
  128B  || 44 || 5  ||  9.09% ||  62.50 %   || 0ms
  128B  || 48 || 1  ||  0.00% ||  62.50 %   || 0ms
  128B  || 48 || 2  ||  0.00% ||  62.50 %   || 0ms
  128B  || 48 || 3  ||  0.00% ||  62.50 %   || 0ms
  128B  || 48 || 4  ||  0.00% ||  62.50 %   || 0ms
  128B  || 48 || 5  ||  0.00% ||  62.50 %   || 0ms
  128B  || 52 || 1  ||  7.69% ||  56.25 %   || 0ms
  128B  || 52 || 2  ||  7.69% ||  56.25 %   || 0ms
  128B  || 52 || 3  ||  7.69% ||  56.25 %   || 0ms
  128B  || 52 || 4  ||  7.69% ||  56.25 %   || 0ms
  128B  || 52 || 5  ||  7.69% ||  56.25 %   || 0ms
  128B  || 56 || 1  ||  0.00% ||  56.25 %   || 0ms
  128B  || 56 || 2  ||  0.00% ||  56.25 %   || 0ms
  128B  || 56 || 3  ||  0.00% ||  56.25 %   || 0ms
  128B  || 56 || 4  ||  0.00% ||  56.25 %   || 0ms
  128B  || 56 || 5  ||  0.00% ||  56.25 %   || 0ms
  128B  || 60 || 1  ||  6.67% ||  50.00 %   || 0ms
  128B  || 60 || 2  ||  6.67% ||  50.00 %   || 0ms
  128B  || 60 || 3  ||  6.67% ||  50.00 %   || 0ms
  128B  || 60 || 4  ||  6.67% ||  50.00 %   || 0ms
  128B  || 60 || 5  ||  6.67% ||  50.00 %   || 0ms
  128B  || 64 || 1  ||  0.00% ||  50.00 %   || 0ms
  128B  || 64 || 2  ||  0.00% ||  50.00 %   || 0ms
  128B  || 64 || 3  ||  0.00% ||  50.00 %   || 0ms
  128B  || 64 || 4  ||  0.00% ||  50.00 %   || 0ms
  128B  || 64 || 5  ||  0.00% ||  50.00 %   || 0ms
  256B  || 28 || 1  || 14.29% ||  12.50 %   || 0ms
  256B  || 28 || 2  || 14.29% ||  12.50 %   || 0ms
  256B  || 28 || 3  || 14.29% ||  12.50 %   || 0ms
  256B  || 28 || 4  || 14.29% ||  12.50 %   || 0ms
  256B  || 28 || 5  || 14.29% ||  12.50 %   || 0ms
  256B  || 32 || 1  ||  0.00% ||  12.50 %   || 0ms
```

256B	32	2	0.00%	12.50 %	0ms
256B	32	3	0.00%	12.50 %	0ms
256B	32	4	0.00%	12.50 %	0ms
256B	32	5	0.00%	12.50 %	0ms
256B	36	1	11.11%	21.88 %	0ms
256B	36	2	11.11%	21.88 %	0ms
256B	36	3	11.11%	21.88 %	0ms
256B	36	4	11.11%	21.88 %	0ms
256B	36	5	11.11%	21.88 %	0ms
256B	40	1	0.00%	21.88 %	0ms
256B	40	2	0.00%	21.88 %	0ms
256B	40	3	0.00%	21.88 %	0ms
256B	40	4	0.00%	21.88 %	0ms
256B	40	5	0.00%	21.88 %	0ms
256B	44	1	9.09%	25.00 %	0ms
256B	44	2	9.09%	25.00 %	0ms
256B	44	3	9.09%	25.00 %	0ms
256B	44	4	9.09%	25.00 %	0ms
256B	44	5	9.09%	25.00 %	0ms
256B	48	1	0.00%	25.00 %	0ms
256B	48	2	0.00%	25.00 %	0ms
256B	48	3	0.00%	25.00 %	0ms
256B	48	4	0.00%	25.00 %	0ms
256B	48	5	0.00%	25.00 %	0ms
256B	52	1	7.69%	34.38 %	0ms
256B	52	2	7.69%	34.38 %	0ms
256B	52	3	7.69%	34.38 %	0ms
256B	52	4	7.69%	34.38 %	0ms
256B	52	5	7.69%	34.38 %	0ms
256B	56	1	0.00%	34.38 %	0ms
256B	56	2	0.00%	34.38 %	0ms
256B	56	3	0.00%	34.38 %	0ms
256B	56	4	0.00%	34.38 %	0ms
256B	56	5	0.00%	34.38 %	0ms
256B	60	1	6.67%	25.00 %	0ms
256B	60	2	6.67%	25.00 %	0ms
256B	60	3	6.67%	25.00 %	0ms
256B	60	4	6.67%	25.00 %	0ms
256B	60	5	6.67%	25.00 %	0ms
256B	64	1	0.00%	25.00 %	0ms
256B	64	2	0.00%	25.00 %	0ms
256B	64	3	0.00%	25.00 %	0ms
256B	64	4	0.00%	25.00 %	0ms
256B	64	5	0.00%	25.00 %	0ms
512B	28	1	14.29%	6.25 %	0ms
512B	28	2	14.29%	6.25 %	0ms
512B	28	3	14.29%	6.25 %	0ms
512B	28	4	14.29%	6.25 %	0ms
512B	28	5	14.29%	6.25 %	0ms
512B	32	1	0.00%	6.25 %	0ms
512B	32	2	0.00%	6.25 %	0ms
512B	32	3	0.00%	6.25 %	0ms
512B	32	4	0.00%	6.25 %	0ms
512B	32	5	0.00%	6.25 %	0ms
512B	36	1	11.11%	14.06 %	0ms
512B	36	2	11.11%	14.06 %	0ms
512B	36	3	11.11%	14.06 %	0ms
512B	36	4	11.11%	14.06 %	0ms
512B	36	5	11.11%	14.06 %	0ms
512B	40	1	0.00%	14.06 %	0ms
512B	40	2	0.00%	14.06 %	0ms
512B	40	3	0.00%	14.06 %	0ms
512B	40	4	0.00%	14.06 %	0ms
512B	40	5	0.00%	14.06 %	0ms
512B	44	1	9.09%	15.63 %	0ms

```
 512B || 44 || 2 ||  9.09% ||  15.63 % || 0ms
 512B || 44 || 3 ||  9.09% ||  15.63 % || 0ms
 512B || 44 || 4 ||  9.09% ||  15.63 % || 0ms
 512B || 44 || 5 ||  9.09% ||  15.63 % || 0ms
 512B || 48 || 1 ||  0.00% ||  15.63 % || 0ms
 512B || 48 || 2 ||  0.00% ||  15.63 % || 0ms
 512B || 48 || 3 ||  0.00% ||  15.63 % || 0ms
 512B || 48 || 4 ||  0.00% ||  15.63 % || 0ms
 512B || 48 || 5 ||  0.00% ||  15.63 % || 0ms
 512B || 52 || 1 ||  7.69% ||  12.50 % || 0ms
 512B || 52 || 2 ||  7.69% ||  12.50 % || 0ms
 512B || 52 || 3 ||  7.69% ||  12.50 % || 0ms
 512B || 52 || 4 ||  7.69% ||  12.50 % || 0ms
 512B || 52 || 5 ||  7.69% ||  12.50 % || 0ms
 512B || 56 || 1 ||  0.00% ||  12.50 % || 0ms
 512B || 56 || 2 ||  0.00% ||  12.50 % || 0ms
 512B || 56 || 3 ||  0.00% ||  12.50 % || 0ms
 512B || 56 || 4 ||  0.00% ||  12.50 % || 0ms
 512B || 56 || 5 ||  0.00% ||  12.50 % || 0ms
 512B || 60 || 1 ||  6.67% ||  12.50 % || 0ms
 512B || 60 || 2 ||  6.67% ||  12.50 % || 0ms
 512B || 60 || 3 ||  6.67% ||  12.50 % || 0ms
 512B || 60 || 4 ||  6.67% ||  12.50 % || 0ms
 512B || 60 || 5 ||  6.67% ||  12.50 % || 0ms
 512B || 64 || 1 ||  0.00% ||  12.50 % || 0ms
 512B || 64 || 2 ||  0.00% ||  12.50 % || 0ms
 512B || 64 || 3 ||  0.00% ||  12.50 % || 0ms
 512B || 64 || 4 ||  0.00% ||  12.50 % || 0ms
 512B || 64 || 5 ||  0.00% ||  12.50 % || 0ms
1024B || 28 || 1 || 14.29% ||   3.13 % || 0ms
1024B || 28 || 2 || 14.29% ||   3.13 % || 0ms
1024B || 28 || 3 || 14.29% ||   3.13 % || 0ms
1024B || 28 || 4 || 14.29% ||   3.13 % || 0ms
1024B || 28 || 5 || 14.29% ||   3.13 % || 0ms
1024B || 32 || 1 ||  0.00% ||   3.13 % || 0ms
1024B || 32 || 2 ||  0.00% ||   3.13 % || 0ms
1024B || 32 || 3 ||  0.00% ||   3.13 % || 0ms
1024B || 32 || 4 ||  0.00% ||   3.13 % || 0ms
1024B || 32 || 5 ||  0.00% ||   3.13 % || 0ms
1024B || 36 || 1 || 11.11% ||   6.25 % || 0ms
1024B || 36 || 2 || 11.11% ||   6.25 % || 0ms
1024B || 36 || 3 || 11.11% ||   6.25 % || 0ms
1024B || 36 || 4 || 11.11% ||   6.25 % || 0ms
1024B || 36 || 5 || 11.11% ||   6.25 % || 0ms
1024B || 40 || 1 ||  0.00% ||   6.25 % || 0ms
1024B || 40 || 2 ||  0.00% ||   6.25 % || 0ms
1024B || 40 || 3 ||  0.00% ||   6.25 % || 0ms
1024B || 40 || 4 ||  0.00% ||   6.25 % || 0ms
1024B || 40 || 5 ||  0.00% ||   6.25 % || 0ms
1024B || 44 || 1 ||  9.09% ||   6.25 % || 0ms
1024B || 44 || 2 ||  9.09% ||   6.25 % || 0ms
1024B || 44 || 3 ||  9.09% ||   6.25 % || 0ms
1024B || 44 || 4 ||  9.09% ||   6.25 % || 0ms
1024B || 44 || 5 ||  9.09% ||   6.25 % || 0ms
1024B || 48 || 1 ||  0.00% ||   6.25 % || 0ms
1024B || 48 || 2 ||  0.00% ||   6.25 % || 0ms
1024B || 48 || 3 ||  0.00% ||   6.25 % || 0ms
1024B || 48 || 4 ||  0.00% ||   6.25 % || 0ms
1024B || 48 || 5 ||  0.00% ||   6.25 % || 0ms
1024B || 52 || 1 ||  7.69% ||   7.03 % || 0ms
1024B || 52 || 2 ||  7.69% ||   7.03 % || 0ms
1024B || 52 || 3 ||  7.69% ||   7.03 % || 0ms
1024B || 52 || 4 ||  7.69% ||   7.03 % || 0ms
1024B || 52 || 5 ||  7.69% ||   7.03 % || 0ms
1024B || 56 || 1 ||  0.00% ||   7.03 % || 0ms
```

1024B	56	2	0.00%	7.03 %	0ms
1024B	56	3	0.00%	7.03 %	0ms
1024B	56	4	0.00%	7.03 %	0ms
1024B	56	5	0.00%	7.03 %	0ms
1024B	60	1	6.67%	6.25 %	0ms
1024B	60	2	6.67%	6.25 %	0ms
1024B	60	3	6.67%	6.25 %	0ms
1024B	60	4	6.67%	6.25 %	0ms
1024B	60	5	6.67%	6.25 %	0ms
1024B	64	1	0.00%	6.25 %	0ms
1024B	64	2	0.00%	6.25 %	0ms
1024B	64	3	0.00%	6.25 %	0ms
1024B	64	4	0.00%	6.25 %	0ms
1024B	64	5	0.00%	6.25 %	0ms
2.00KB	28	1	14.29%	1.56 %	0ms
2.00KB	28	2	14.29%	1.56 %	0ms
2.00KB	28	3	14.29%	1.56 %	0ms
2.00KB	28	4	14.29%	1.56 %	0ms
2.00KB	28	5	14.29%	1.56 %	0ms
2.00KB	32	1	0.00%	1.56 %	0ms
2.00KB	32	2	0.00%	1.56 %	0ms
2.00KB	32	3	0.00%	1.56 %	0ms
2.00KB	32	4	0.00%	1.56 %	0ms
2.00KB	32	5	0.00%	1.56 %	0ms
2.00KB	36	1	11.11%	2.34 %	0ms
2.00KB	36	2	11.11%	2.34 %	0ms
2.00KB	36	3	11.11%	2.34 %	0ms
2.00KB	36	4	11.11%	2.34 %	0ms
2.00KB	36	5	11.11%	2.34 %	0ms
2.00KB	40	1	0.00%	2.34 %	0ms
2.00KB	40	2	0.00%	2.34 %	0ms
2.00KB	40	3	0.00%	2.34 %	0ms
2.00KB	40	4	0.00%	2.34 %	0ms
2.00KB	40	5	0.00%	2.34 %	0ms
2.00KB	44	1	9.09%	3.91 %	0ms
2.00KB	44	2	9.09%	3.91 %	0ms
2.00KB	44	3	9.09%	3.91 %	0ms
2.00KB	44	4	9.09%	3.91 %	0ms
2.00KB	44	5	9.09%	3.91 %	0ms
2.00KB	48	1	0.00%	3.91 %	0ms
2.00KB	48	2	0.00%	3.91 %	0ms
2.00KB	48	3	0.00%	3.91 %	0ms
2.00KB	48	4	0.00%	3.91 %	0ms
2.00KB	48	5	0.00%	3.91 %	0ms
2.00KB	52	1	7.69%	4.30 %	0ms
2.00KB	52	2	7.69%	4.30 %	0ms
2.00KB	52	3	7.69%	4.30 %	0ms
2.00KB	52	4	7.69%	4.30 %	0ms
2.00KB	52	5	7.69%	4.30 %	0ms
2.00KB	56	1	0.00%	4.30 %	0ms
2.00KB	56	2	0.00%	4.30 %	0ms
2.00KB	56	3	0.00%	4.30 %	0ms
2.00KB	56	4	0.00%	4.30 %	0ms
2.00KB	56	5	0.00%	4.30 %	0ms
2.00KB	60	1	6.67%	3.13 %	0ms
2.00KB	60	2	6.67%	3.13 %	0ms
2.00KB	60	3	6.67%	3.13 %	0ms
2.00KB	60	4	6.67%	3.13 %	0ms
2.00KB	60	5	6.67%	3.13 %	0ms
2.00KB	64	1	0.00%	3.13 %	0ms
2.00KB	64	2	0.00%	3.13 %	0ms
2.00KB	64	3	0.00%	3.13 %	0ms
2.00KB	64	4	0.00%	3.13 %	0ms
2.00KB	64	5	0.00%	3.13 %	0ms
4.00KB	28	1	14.29%	0.78 %	0ms

```
   4.00KB || 28 || 2  || 14.29% ||   0.78 %   || 0ms
   4.00KB || 28 || 3  || 14.29% ||   0.78 %   || 0ms
   4.00KB || 28 || 4  || 14.29% ||   0.78 %   || 0ms
   4.00KB || 28 || 5  || 14.29% ||   0.78 %   || 0ms
   4.00KB || 32 || 1  ||  0.00% ||   0.78 %   || 0ms
   4.00KB || 32 || 2  ||  0.00% ||   0.78 %   || 0ms
   4.00KB || 32 || 3  ||  0.00% ||   0.78 %   || 0ms
   4.00KB || 32 || 4  ||  0.00% ||   0.78 %   || 0ms
   4.00KB || 32 || 5  ||  0.00% ||   0.78 %   || 0ms
   4.00KB || 36 || 1  || 11.11% ||   1.37 %   || 0ms
   4.00KB || 36 || 2  || 11.11% ||   1.37 %   || 0ms
   4.00KB || 36 || 3  || 11.11% ||   1.37 %   || 0ms
   4.00KB || 36 || 4  || 11.11% ||   1.37 %   || 0ms
   4.00KB || 36 || 5  || 11.11% ||   1.37 %   || 0ms
   4.00KB || 40 || 1  ||  0.00% ||   1.37 %   || 0ms
   4.00KB || 40 || 2  ||  0.00% ||   1.37 %   || 0ms
   4.00KB || 40 || 3  ||  0.00% ||   1.37 %   || 0ms
   4.00KB || 40 || 4  ||  0.00% ||   1.37 %   || 0ms
   4.00KB || 40 || 5  ||  0.00% ||   1.37 %   || 0ms
   4.00KB || 44 || 1  ||  9.09% ||   1.56 %   || 0ms
   4.00KB || 44 || 2  ||  9.09% ||   1.56 %   || 0ms
   4.00KB || 44 || 3  ||  9.09% ||   1.56 %   || 0ms
   4.00KB || 44 || 4  ||  9.09% ||   1.56 %   || 0ms
   4.00KB || 44 || 5  ||  9.09% ||   1.56 %   || 0ms
   4.00KB || 48 || 1  ||  0.00% ||   1.56 %   || 0ms
   4.00KB || 48 || 2  ||  0.00% ||   1.56 %   || 0ms
   4.00KB || 48 || 3  ||  0.00% ||   1.56 %   || 0ms
   4.00KB || 48 || 4  ||  0.00% ||   1.56 %   || 0ms
   4.00KB || 48 || 5  ||  0.00% ||   1.56 %   || 0ms
   4.00KB || 52 || 1  ||  7.69% ||   1.56 %   || 0ms
   4.00KB || 52 || 2  ||  7.69% ||   1.56 %   || 0ms
   4.00KB || 52 || 3  ||  7.69% ||   1.56 %   || 0ms
   4.00KB || 52 || 4  ||  7.69% ||   1.56 %   || 0ms
   4.00KB || 52 || 5  ||  7.69% ||   1.56 %   || 0ms
   4.00KB || 56 || 1  ||  0.00% ||   1.56 %   || 0ms
   4.00KB || 56 || 2  ||  0.00% ||   1.56 %   || 0ms
   4.00KB || 56 || 3  ||  0.00% ||   1.56 %   || 0ms
   4.00KB || 56 || 4  ||  0.00% ||   1.56 %   || 0ms
   4.00KB || 56 || 5  ||  0.00% ||   1.56 %   || 0ms
   4.00KB || 60 || 1  ||  6.67% ||   1.56 %   || 0ms
   4.00KB || 60 || 2  ||  6.67% ||   1.56 %   || 0ms
   4.00KB || 60 || 3  ||  6.67% ||   1.56 %   || 0ms
   4.00KB || 60 || 4  ||  6.67% ||   1.56 %   || 0ms
   4.00KB || 60 || 5  ||  6.67% ||   1.56 %   || 0ms
   4.00KB || 64 || 1  ||  0.00% ||   1.56 %   || 0ms
   4.00KB || 64 || 2  ||  0.00% ||   1.56 %   || 0ms
   4.00KB || 64 || 3  ||  0.00% ||   1.56 %   || 0ms
   4.00KB || 64 || 4  ||  0.00% ||   1.56 %   || 0ms
   4.00KB || 64 || 5  ||  0.00% ||   1.56 %   || 0ms
   8.00KB || 28 || 1  || 14.29% ||   0.39 %   || 0ms
   8.00KB || 28 || 2  || 14.29% ||   0.39 %   || 0ms
   8.00KB || 28 || 3  || 14.29% ||   0.39 %   || 0ms
   8.00KB || 28 || 4  || 14.29% ||   0.39 %   || 0ms
   8.00KB || 28 || 5  || 14.29% ||   0.39 %   || 0ms
   8.00KB || 32 || 1  ||  0.00% ||   0.39 %   || 0ms
   8.00KB || 32 || 2  ||  0.00% ||   0.39 %   || 0ms
   8.00KB || 32 || 3  ||  0.00% ||   0.39 %   || 0ms
   8.00KB || 32 || 4  ||  0.00% ||   0.39 %   || 0ms
   8.00KB || 32 || 5  ||  0.00% ||   0.39 %   || 0ms
   8.00KB || 36 || 1  || 11.11% ||   0.88 %   || 0ms
   8.00KB || 36 || 2  || 11.11% ||   0.88 %   || 0ms
   8.00KB || 36 || 3  || 11.11% ||   0.88 %   || 0ms
   8.00KB || 36 || 4  || 11.11% ||   0.88 %   || 0ms
   8.00KB || 36 || 5  || 11.11% ||   0.88 %   || 0ms
   8.00KB || 40 || 1  ||  0.00% ||   0.88 %   || 0ms
```

```
   8.00KB || 40 || 2  ||  0.00% ||   0.88 %   || 0ms
   8.00KB || 40 || 3  ||  0.00% ||   0.88 %   || 0ms
   8.00KB || 40 || 4  ||  0.00% ||   0.88 %   || 0ms
   8.00KB || 40 || 5  ||  0.00% ||   0.88 %   || 0ms
   8.00KB || 44 || 1  ||  9.09% ||   0.98 %   || 0ms
   8.00KB || 44 || 2  ||  9.09% ||   0.98 %   || 0ms
   8.00KB || 44 || 3  ||  9.09% ||   0.98 %   || 0ms
   8.00KB || 44 || 4  ||  9.09% ||   0.98 %   || 0ms
   8.00KB || 44 || 5  ||  9.09% ||   0.98 %   || 0ms
   8.00KB || 48 || 1  ||  0.00% ||   0.98 %   || 0ms
   8.00KB || 48 || 2  ||  0.00% ||   0.98 %   || 0ms
   8.00KB || 48 || 3  ||  0.00% ||   0.98 %   || 0ms
   8.00KB || 48 || 4  ||  0.00% ||   0.98 %   || 0ms
   8.00KB || 48 || 5  ||  0.00% ||   0.98 %   || 0ms
   8.00KB || 52 || 1  ||  7.69% ||   0.88 %   || 0ms
   8.00KB || 52 || 2  ||  7.69% ||   0.88 %   || 0ms
   8.00KB || 52 || 3  ||  7.69% ||   0.88 %   || 0ms
   8.00KB || 52 || 4  ||  7.69% ||   0.88 %   || 0ms
   8.00KB || 52 || 5  ||  7.69% ||   0.88 %   || 0ms
   8.00KB || 56 || 1  ||  0.00% ||   0.88 %   || 0ms
   8.00KB || 56 || 2  ||  0.00% ||   0.88 %   || 0ms
   8.00KB || 56 || 3  ||  0.00% ||   0.88 %   || 0ms
   8.00KB || 56 || 4  ||  0.00% ||   0.88 %   || 0ms
   8.00KB || 56 || 5  ||  0.00% ||   0.88 %   || 0ms
   8.00KB || 60 || 1  ||  6.67% ||   0.78 %   || 0ms
   8.00KB || 60 || 2  ||  6.67% ||   0.78 %   || 0ms
   8.00KB || 60 || 3  ||  6.67% ||   0.78 %   || 0ms
   8.00KB || 60 || 4  ||  6.67% ||   0.78 %   || 0ms
   8.00KB || 60 || 5  ||  6.67% ||   0.78 %   || 0ms
   8.00KB || 64 || 1  ||  0.00% ||   0.78 %   || 0ms
   8.00KB || 64 || 2  ||  0.00% ||   0.78 %   || 0ms
   8.00KB || 64 || 3  ||  0.00% ||   0.78 %   || 0ms
   8.00KB || 64 || 4  ||  0.00% ||   0.78 %   || 0ms
   8.00KB || 64 || 5  ||  0.00% ||   0.78 %   || 0ms
  16.00KB || 28 || 1  || 14.29% ||   0.20 %   || 0ms
  16.00KB || 28 || 2  || 14.29% ||   0.20 %   || 0ms
  16.00KB || 28 || 3  || 14.29% ||   0.20 %   || 0ms
  16.00KB || 28 || 4  || 14.29% ||   0.20 %   || 0ms
  16.00KB || 28 || 5  || 14.29% ||   0.20 %   || 0ms
  16.00KB || 32 || 1  ||  0.00% ||   0.20 %   || 0ms
  16.00KB || 32 || 2  ||  0.00% ||   0.20 %   || 0ms
  16.00KB || 32 || 3  ||  0.00% ||   0.20 %   || 0ms
  16.00KB || 32 || 4  ||  0.00% ||   0.20 %   || 0ms
  16.00KB || 32 || 5  ||  0.00% ||   0.20 %   || 0ms
  16.00KB || 36 || 1  || 11.11% ||   0.39 %   || 0ms
  16.00KB || 36 || 2  || 11.11% ||   0.39 %   || 0ms
  16.00KB || 36 || 3  || 11.11% ||   0.39 %   || 0ms
  16.00KB || 36 || 4  || 11.11% ||   0.39 %   || 0ms
  16.00KB || 36 || 5  || 11.11% ||   0.39 %   || 0ms
  16.00KB || 40 || 1  ||  0.00% ||   0.39 %   || 0ms
  16.00KB || 40 || 2  ||  0.00% ||   0.39 %   || 0ms
  16.00KB || 40 || 3  ||  0.00% ||   0.39 %   || 0ms
  16.00KB || 40 || 4  ||  0.00% ||   0.39 %   || 0ms
  16.00KB || 40 || 5  ||  0.00% ||   0.39 %   || 0ms
  16.00KB || 44 || 1  ||  9.09% ||   0.39 %   || 0ms
  16.00KB || 44 || 2  ||  9.09% ||   0.39 %   || 0ms
  16.00KB || 44 || 3  ||  9.09% ||   0.39 %   || 0ms
  16.00KB || 44 || 4  ||  9.09% ||   0.39 %   || 0ms
  16.00KB || 44 || 5  ||  9.09% ||   0.39 %   || 0ms
  16.00KB || 48 || 1  ||  0.00% ||   0.39 %   || 0ms
  16.00KB || 48 || 2  ||  0.00% ||   0.39 %   || 0ms
  16.00KB || 48 || 3  ||  0.00% ||   0.39 %   || 0ms
  16.00KB || 48 || 4  ||  0.00% ||   0.39 %   || 0ms
  16.00KB || 48 || 5  ||  0.00% ||   0.39 %   || 1ms
  16.00KB || 52 || 1  ||  7.69% ||   0.54 %   || 0ms
```

```
  16.00KB || 52 || 2  ||  7.69% ||   0.54 %   || 0ms
  16.00KB || 52 || 3  ||  7.69% ||   0.54 %   || 0ms
  16.00KB || 52 || 4  ||  7.69% ||   0.54 %   || 0ms
  16.00KB || 52 || 5  ||  7.69% ||   0.54 %   || 0ms
  16.00KB || 56 || 1  ||  0.00% ||   0.54 %   || 0ms
  16.00KB || 56 || 2  ||  0.00% ||   0.54 %   || 0ms
  16.00KB || 56 || 3  ||  0.00% ||   0.54 %   || 0ms
  16.00KB || 56 || 4  ||  0.00% ||   0.54 %   || 0ms
  16.00KB || 56 || 5  ||  0.00% ||   0.54 %   || 0ms
  16.00KB || 60 || 1  ||  6.67% ||   0.39 %   || 0ms
  16.00KB || 60 || 2  ||  6.67% ||   0.39 %   || 0ms
  16.00KB || 60 || 3  ||  6.67% ||   0.39 %   || 0ms
  16.00KB || 60 || 4  ||  6.67% ||   0.39 %   || 0ms
  16.00KB || 60 || 5  ||  6.67% ||   0.39 %   || 0ms
  16.00KB || 64 || 1  ||  0.00% ||   0.39 %   || 0ms
  16.00KB || 64 || 2  ||  0.00% ||   0.39 %   || 0ms
  16.00KB || 64 || 3  ||  0.00% ||   0.39 %   || 0ms
  16.00KB || 64 || 4  ||  0.00% ||   0.39 %   || 0ms
  16.00KB || 64 || 5  ||  0.00% ||   0.39 %   || 0ms
  32.00KB || 28 || 1  || 14.29% ||   0.10 %   || 0ms
  32.00KB || 28 || 2  || 14.29% ||   0.10 %   || 0ms
  32.00KB || 28 || 3  || 14.29% ||   0.10 %   || 0ms
  32.00KB || 28 || 4  || 14.29% ||   0.10 %   || 0ms
  32.00KB || 28 || 5  || 14.29% ||   0.10 %   || 0ms
  32.00KB || 32 || 1  ||  0.00% ||   0.10 %   || 0ms
  32.00KB || 32 || 2  ||  0.00% ||   0.10 %   || 0ms
  32.00KB || 32 || 3  ||  0.00% ||   0.10 %   || 0ms
  32.00KB || 32 || 4  ||  0.00% ||   0.10 %   || 0ms
  32.00KB || 32 || 5  ||  0.00% ||   0.10 %   || 0ms
  32.00KB || 36 || 1  || 11.11% ||   0.15 %   || 0ms
  32.00KB || 36 || 2  || 11.11% ||   0.15 %   || 0ms
  32.00KB || 36 || 3  || 11.11% ||   0.15 %   || 0ms
  32.00KB || 36 || 4  || 11.11% ||   0.15 %   || 0ms
  32.00KB || 36 || 5  || 11.11% ||   0.15 %   || 0ms
  32.00KB || 40 || 1  ||  0.00% ||   0.15 %   || 0ms
  32.00KB || 40 || 2  ||  0.00% ||   0.15 %   || 0ms
  32.00KB || 40 || 3  ||  0.00% ||   0.15 %   || 0ms
  32.00KB || 40 || 4  ||  0.00% ||   0.15 %   || 1ms
  32.00KB || 40 || 5  ||  0.00% ||   0.15 %   || 0ms
  32.00KB || 44 || 1  ||  9.09% ||   0.24 %   || 0ms
  32.00KB || 44 || 2  ||  9.09% ||   0.24 %   || 0ms
  32.00KB || 44 || 3  ||  9.09% ||   0.24 %   || 0ms
  32.00KB || 44 || 4  ||  9.09% ||   0.24 %   || 0ms
  32.00KB || 44 || 5  ||  9.09% ||   0.24 %   || 0ms
  32.00KB || 48 || 1  ||  0.00% ||   0.24 %   || 0ms
  32.00KB || 48 || 2  ||  0.00% ||   0.24 %   || 0ms
  32.00KB || 48 || 3  ||  0.00% ||   0.24 %   || 0ms
  32.00KB || 48 || 4  ||  0.00% ||   0.24 %   || 0ms
  32.00KB || 48 || 5  ||  0.00% ||   0.24 %   || 0ms
  32.00KB || 52 || 1  ||  7.69% ||   0.20 %   || 0ms
  32.00KB || 52 || 2  ||  7.69% ||   0.20 %   || 0ms
  32.00KB || 52 || 3  ||  7.69% ||   0.20 %   || 0ms
  32.00KB || 52 || 4  ||  7.69% ||   0.20 %   || 0ms
  32.00KB || 52 || 5  ||  7.69% ||   0.20 %   || 0ms
  32.00KB || 56 || 1  ||  0.00% ||   0.20 %   || 0ms
  32.00KB || 56 || 2  ||  0.00% ||   0.20 %   || 0ms
  32.00KB || 56 || 3  ||  0.00% ||   0.20 %   || 0ms
  32.00KB || 56 || 4  ||  0.00% ||   0.20 %   || 0ms
  32.00KB || 56 || 5  ||  0.00% ||   0.20 %   || 0ms
  32.00KB || 60 || 1  ||  6.67% ||   0.20 %   || 0ms
  32.00KB || 60 || 2  ||  6.67% ||   0.20 %   || 0ms
  32.00KB || 60 || 3  ||  6.67% ||   0.20 %   || 0ms
  32.00KB || 60 || 4  ||  6.67% ||   0.20 %   || 0ms
  32.00KB || 60 || 5  ||  6.67% ||   0.20 %   || 0ms
  32.00KB || 64 || 1  ||  0.00% ||   0.20 %   || 0ms
```

```
  32.00KB || 64 || 2  ||  0.00% ||   0.20 %   || 0ms
  32.00KB || 64 || 3  ||  0.00% ||   0.20 %   || 1ms
  32.00KB || 64 || 4  ||  0.00% ||   0.20 %   || 0ms
  32.00KB || 64 || 5  ||  0.00% ||   0.20 %   || 0ms
  64.00KB || 28 || 1  || 14.29% ||   0.05 %   || 0ms
  64.00KB || 28 || 2  || 14.29% ||   0.05 %   || 0ms
  64.00KB || 28 || 3  || 14.29% ||   0.05 %   || 0ms
  64.00KB || 28 || 4  || 14.29% ||   0.05 %   || 0ms
  64.00KB || 28 || 5  || 14.29% ||   0.05 %   || 0ms
  64.00KB || 32 || 1  ||  0.00% ||   0.05 %   || 0ms
  64.00KB || 32 || 2  ||  0.00% ||   0.05 %   || 0ms
  64.00KB || 32 || 3  ||  0.00% ||   0.05 %   || 0ms
  64.00KB || 32 || 4  ||  0.00% ||   0.05 %   || 0ms
  64.00KB || 32 || 5  ||  0.00% ||   0.05 %   || 0ms
  64.00KB || 36 || 1  || 11.11% ||   0.09 %   || 0ms
  64.00KB || 36 || 2  || 11.11% ||   0.09 %   || 0ms
  64.00KB || 36 || 3  || 11.11% ||   0.09 %   || 0ms
  64.00KB || 36 || 4  || 11.11% ||   0.09 %   || 0ms
  64.00KB || 36 || 5  || 11.11% ||   0.09 %   || 0ms
  64.00KB || 40 || 1  ||  0.00% ||   0.09 %   || 0ms
  64.00KB || 40 || 2  ||  0.00% ||   0.09 %   || 1ms
  64.00KB || 40 || 3  ||  0.00% ||   0.09 %   || 0ms
  64.00KB || 40 || 4  ||  0.00% ||   0.09 %   || 0ms
  64.00KB || 40 || 5  ||  0.00% ||   0.09 %   || 0ms
  64.00KB || 44 || 1  ||  9.09% ||   0.10 %   || 0ms
  64.00KB || 44 || 2  ||  9.09% ||   0.10 %   || 0ms
  64.00KB || 44 || 3  ||  9.09% ||   0.10 %   || 0ms
  64.00KB || 44 || 4  ||  9.09% ||   0.10 %   || 0ms
  64.00KB || 44 || 5  ||  9.09% ||   0.10 %   || 0ms
  64.00KB || 48 || 1  ||  0.00% ||   0.10 %   || 0ms
  64.00KB || 48 || 2  ||  0.00% ||   0.10 %   || 0ms
  64.00KB || 48 || 3  ||  0.00% ||   0.10 %   || 0ms
  64.00KB || 48 || 4  ||  0.00% ||   0.10 %   || 0ms
  64.00KB || 48 || 5  ||  0.00% ||   0.10 %   || 0ms
  64.00KB || 52 || 1  ||  7.69% ||   0.11 %   || 0ms
  64.00KB || 52 || 2  ||  7.69% ||   0.11 %   || 0ms
  64.00KB || 52 || 3  ||  7.69% ||   0.11 %   || 0ms
  64.00KB || 52 || 4  ||  7.69% ||   0.11 %   || 1ms
  64.00KB || 52 || 5  ||  7.69% ||   0.11 %   || 0ms
  64.00KB || 56 || 1  ||  0.00% ||   0.11 %   || 0ms
  64.00KB || 56 || 2  ||  0.00% ||   0.11 %   || 0ms
  64.00KB || 56 || 3  ||  0.00% ||   0.11 %   || 0ms
  64.00KB || 56 || 4  ||  0.00% ||   0.11 %   || 0ms
  64.00KB || 56 || 5  ||  0.00% ||   0.11 %   || 0ms
  64.00KB || 60 || 1  ||  6.67% ||   0.10 %   || 0ms
  64.00KB || 60 || 2  ||  6.67% ||   0.10 %   || 0ms
  64.00KB || 60 || 3  ||  6.67% ||   0.10 %   || 0ms
  64.00KB || 60 || 4  ||  6.67% ||   0.10 %   || 0ms
  64.00KB || 60 || 5  ||  6.67% ||   0.10 %   || 0ms
  64.00KB || 64 || 1  ||  0.00% ||   0.10 %   || 0ms
  64.00KB || 64 || 2  ||  0.00% ||   0.10 %   || 0ms
  64.00KB || 64 || 3  ||  0.00% ||   0.10 %   || 0ms
  64.00KB || 64 || 4  ||  0.00% ||   0.10 %   || 0ms
  64.00KB || 64 || 5  ||  0.00% ||   0.10 %   || 0ms
 128.00KB || 28 || 1  || 14.29% ||   0.02 %   || 1ms
 128.00KB || 28 || 2  || 14.29% ||   0.02 %   || 0ms
 128.00KB || 28 || 3  || 14.29% ||   0.02 %   || 0ms
 128.00KB || 28 || 4  || 14.29% ||   0.02 %   || 0ms
 128.00KB || 28 || 5  || 14.29% ||   0.02 %   || 0ms
 128.00KB || 32 || 1  ||  0.00% ||   0.02 %   || 0ms
 128.00KB || 32 || 2  ||  0.00% ||   0.02 %   || 0ms
 128.00KB || 32 || 3  ||  0.00% ||   0.02 %   || 0ms
 128.00KB || 32 || 4  ||  0.00% ||   0.02 %   || 1ms
 128.00KB || 32 || 5  ||  0.00% ||   0.02 %   || 0ms
 128.00KB || 36 || 1  || 11.11% ||   0.05 %   || 0ms
```

```
 128.00KB || 36 || 2  || 11.11% ||   0.05 %   || 0ms
 128.00KB || 36 || 3  || 11.11% ||   0.05 %   || 0ms
 128.00KB || 36 || 4  || 11.11% ||   0.05 %   || 0ms
 128.00KB || 36 || 5  || 11.11% ||   0.05 %   || 0ms
 128.00KB || 40 || 1  ||  0.00% ||   0.05 %   || 0ms
 128.00KB || 40 || 2  ||  0.00% ||   0.05 %   || 0ms
 128.00KB || 40 || 3  ||  0.00% ||   0.05 %   || 0ms
 128.00KB || 40 || 4  ||  0.00% ||   0.05 %   || 0ms
 128.00KB || 40 || 5  ||  0.00% ||   0.05 %   || 1ms
 128.00KB || 44 || 1  ||  9.09% ||   0.06 %   || 0ms
 128.00KB || 44 || 2  ||  9.09% ||   0.06 %   || 0ms
 128.00KB || 44 || 3  ||  9.09% ||   0.06 %   || 0ms
 128.00KB || 44 || 4  ||  9.09% ||   0.06 %   || 0ms
 128.00KB || 44 || 5  ||  9.09% ||   0.06 %   || 0ms
 128.00KB || 48 || 1  ||  0.00% ||   0.06 %   || 0ms
 128.00KB || 48 || 2  ||  0.00% ||   0.06 %   || 0ms
 128.00KB || 48 || 3  ||  0.00% ||   0.06 %   || 0ms
 128.00KB || 48 || 4  ||  0.00% ||   0.06 %   || 0ms
 128.00KB || 48 || 5  ||  0.00% ||   0.06 %   || 0ms
 128.00KB || 52 || 1  ||  7.69% ||   0.07 %   || 0ms
 128.00KB || 52 || 2  ||  7.69% ||   0.07 %   || 0ms
 128.00KB || 52 || 3  ||  7.69% ||   0.07 %   || 0ms
 128.00KB || 52 || 4  ||  7.69% ||   0.07 %   || 1ms
 128.00KB || 52 || 5  ||  7.69% ||   0.07 %   || 0ms
 128.00KB || 56 || 1  ||  0.00% ||   0.07 %   || 0ms
 128.00KB || 56 || 2  ||  0.00% ||   0.07 %   || 0ms
 128.00KB || 56 || 3  ||  0.00% ||   0.07 %   || 0ms
 128.00KB || 56 || 4  ||  0.00% ||   0.07 %   || 0ms
 128.00KB || 56 || 5  ||  0.00% ||   0.07 %   || 0ms
 128.00KB || 60 || 1  ||  6.67% ||   0.05 %   || 0ms
 128.00KB || 60 || 2  ||  6.67% ||   0.05 %   || 0ms
 128.00KB || 60 || 3  ||  6.67% ||   0.05 %   || 0ms
 128.00KB || 60 || 4  ||  6.67% ||   0.05 %   || 0ms
 128.00KB || 60 || 5  ||  6.67% ||   0.05 %   || 0ms
 128.00KB || 64 || 1  ||  0.00% ||   0.05 %   || 0ms
 128.00KB || 64 || 2  ||  0.00% ||   0.05 %   || 0ms
 128.00KB || 64 || 3  ||  0.00% ||   0.05 %   || 0ms
 128.00KB || 64 || 4  ||  0.00% ||   0.05 %   || 0ms
 128.00KB || 64 || 5  ||  0.00% ||   0.05 %   || 1ms
 256.00KB || 28 || 1  || 14.29% ||   0.01 %   || 0ms
 256.00KB || 28 || 2  || 14.29% ||   0.01 %   || 0ms
 256.00KB || 28 || 3  || 14.29% ||   0.01 %   || 0ms
 256.00KB || 28 || 4  || 14.29% ||   0.01 %   || 0ms
 256.00KB || 28 || 5  || 14.29% ||   0.01 %   || 1ms
 256.00KB || 32 || 1  ||  0.00% ||   0.01 %   || 0ms
 256.00KB || 32 || 2  ||  0.00% ||   0.01 %   || 0ms
 256.00KB || 32 || 3  ||  0.00% ||   0.01 %   || 0ms
 256.00KB || 32 || 4  ||  0.00% ||   0.01 %   || 1ms
 256.00KB || 32 || 5  ||  0.00% ||   0.01 %   || 0ms
 256.00KB || 36 || 1  || 11.11% ||   0.02 %   || 0ms
 256.00KB || 36 || 2  || 11.11% ||   0.02 %   || 0ms
 256.00KB || 36 || 3  || 11.11% ||   0.02 %   || 0ms
 256.00KB || 36 || 4  || 11.11% ||   0.02 %   || 1ms
 256.00KB || 36 || 5  || 11.11% ||   0.02 %   || 0ms
 256.00KB || 40 || 1  ||  0.00% ||   0.02 %   || 0ms
 256.00KB || 40 || 2  ||  0.00% ||   0.02 %   || 0ms
 256.00KB || 40 || 3  ||  0.00% ||   0.02 %   || 0ms
 256.00KB || 40 || 4  ||  0.00% ||   0.02 %   || 1ms
 256.00KB || 40 || 5  ||  0.00% ||   0.02 %   || 0ms
 256.00KB || 44 || 1  ||  9.09% ||   0.02 %   || 0ms
 256.00KB || 44 || 2  ||  9.09% ||   0.02 %   || 0ms
 256.00KB || 44 || 3  ||  9.09% ||   0.02 %   || 0ms
 256.00KB || 44 || 4  ||  9.09% ||   0.02 %   || 0ms
 256.00KB || 44 || 5  ||  9.09% ||   0.02 %   || 0ms
 256.00KB || 48 || 1  ||  0.00% ||   0.02 %   || 1ms
```

```
 256.00KB || 48 || 2  ||  0.00% ||   0.02 %   || 0ms
 256.00KB || 48 || 3  ||  0.00% ||   0.02 %   || 0ms
 256.00KB || 48 || 4  ||  0.00% ||   0.02 %   || 0ms
 256.00KB || 48 || 5  ||  0.00% ||   0.02 %   || 0ms
 256.00KB || 52 || 1  ||  7.69% ||   0.02 %   || 0ms
 256.00KB || 52 || 2  ||  7.69% ||   0.02 %   || 1ms
 256.00KB || 52 || 3  ||  7.69% ||   0.02 %   || 0ms
 256.00KB || 52 || 4  ||  7.69% ||   0.02 %   || 0ms
 256.00KB || 52 || 5  ||  7.69% ||   0.02 %   || 0ms
 256.00KB || 56 || 1  ||  0.00% ||   0.02 %   || 0ms
 256.00KB || 56 || 2  ||  0.00% ||   0.02 %   || 0ms
 256.00KB || 56 || 3  ||  0.00% ||   0.02 %   || 0ms
 256.00KB || 56 || 4  ||  0.00% ||   0.02 %   || 0ms
 256.00KB || 56 || 5  ||  0.00% ||   0.02 %   || 1ms
 256.00KB || 60 || 1  ||  6.67% ||   0.02 %   || 0ms
 256.00KB || 60 || 2  ||  6.67% ||   0.02 %   || 0ms
 256.00KB || 60 || 3  ||  6.67% ||   0.02 %   || 0ms
 256.00KB || 60 || 4  ||  6.67% ||   0.02 %   || 0ms
 256.00KB || 60 || 5  ||  6.67% ||   0.02 %   || 0ms
 256.00KB || 64 || 1  ||  0.00% ||   0.02 %   || 0ms
 256.00KB || 64 || 2  ||  0.00% ||   0.02 %   || 0ms
 256.00KB || 64 || 3  ||  0.00% ||   0.02 %   || 0ms
 256.00KB || 64 || 4  ||  0.00% ||   0.02 %   || 1ms
 256.00KB || 64 || 5  ||  0.00% ||   0.02 %   || 0ms
 512.00KB || 28 || 1  || 14.29% ||   0.01 %   || 0ms
 512.00KB || 28 || 2  || 14.29% ||   0.01 %   || 1ms
 512.00KB || 28 || 3  || 14.29% ||   0.01 %   || 0ms
 512.00KB || 28 || 4  || 14.29% ||   0.01 %   || 1ms
 512.00KB || 28 || 5  || 14.29% ||   0.01 %   || 0ms
 512.00KB || 32 || 1  ||  0.00% ||   0.01 %   || 0ms
 512.00KB || 32 || 2  ||  0.00% ||   0.01 %   || 1ms
 512.00KB || 32 || 3  ||  0.00% ||   0.01 %   || 0ms
 512.00KB || 32 || 4  ||  0.00% ||   0.01 %   || 1ms
 512.00KB || 32 || 5  ||  0.00% ||   0.01 %   || 0ms
 512.00KB || 36 || 1  || 11.11% ||   0.01 %   || 1ms
 512.00KB || 36 || 2  || 11.11% ||   0.01 %   || 0ms
 512.00KB || 36 || 3  || 11.11% ||   0.01 %   || 0ms
 512.00KB || 36 || 4  || 11.11% ||   0.01 %   || 1ms
 512.00KB || 36 || 5  || 11.11% ||   0.01 %   || 0ms
 512.00KB || 40 || 1  ||  0.00% ||   0.01 %   || 0ms
 512.00KB || 40 || 2  ||  0.00% ||   0.01 %   || 1ms
 512.00KB || 40 || 3  ||  0.00% ||   0.01 %   || 0ms
 512.00KB || 40 || 4  ||  0.00% ||   0.01 %   || 0ms
 512.00KB || 40 || 5  ||  0.00% ||   0.01 %   || 1ms
 512.00KB || 44 || 1  ||  9.09% ||   0.02 %   || 0ms
 512.00KB || 44 || 2  ||  9.09% ||   0.02 %   || 0ms
 512.00KB || 44 || 3  ||  9.09% ||   0.02 %   || 1ms
 512.00KB || 44 || 4  ||  9.09% ||   0.02 %   || 0ms
 512.00KB || 44 || 5  ||  9.09% ||   0.02 %   || 0ms
 512.00KB || 48 || 1  ||  0.00% ||   0.02 %   || 1ms
 512.00KB || 48 || 2  ||  0.00% ||   0.02 %   || 0ms
 512.00KB || 48 || 3  ||  0.00% ||   0.02 %   || 0ms
 512.00KB || 48 || 4  ||  0.00% ||   0.02 %   || 0ms
 512.00KB || 48 || 5  ||  0.00% ||   0.02 %   || 1ms
 512.00KB || 52 || 1  ||  7.69% ||   0.01 %   || 0ms
 512.00KB || 52 || 2  ||  7.69% ||   0.01 %   || 0ms
 512.00KB || 52 || 3  ||  7.69% ||   0.01 %   || 1ms
 512.00KB || 52 || 4  ||  7.69% ||   0.01 %   || 0ms
 512.00KB || 52 || 5  ||  7.69% ||   0.01 %   || 0ms
 512.00KB || 56 || 1  ||  0.00% ||   0.01 %   || 0ms
 512.00KB || 56 || 2  ||  0.00% ||   0.01 %   || 1ms
 512.00KB || 56 || 3  ||  0.00% ||   0.01 %   || 0ms
 512.00KB || 56 || 4  ||  0.00% ||   0.01 %   || 0ms
 512.00KB || 56 || 5  ||  0.00% ||   0.01 %   || 0ms
 512.00KB || 60 || 1  ||  6.67% ||   0.01 %   || 1ms
```

```
 512.00KB || 60 || 2  ||  6.67% ||   0.01 %   || 0ms
 512.00KB || 60 || 3  ||  6.67% ||   0.01 %   || 0ms
 512.00KB || 60 || 4  ||  6.67% ||   0.01 %   || 0ms
 512.00KB || 60 || 5  ||  6.67% ||   0.01 %   || 0ms
 512.00KB || 64 || 1  ||  0.00% ||   0.01 %   || 1ms
 512.00KB || 64 || 2  ||  0.00% ||   0.01 %   || 0ms
 512.00KB || 64 || 3  ||  0.00% ||   0.01 %   || 0ms
 512.00KB || 64 || 4  ||  0.00% ||   0.01 %   || 0ms
 512.00KB || 64 || 5  ||  0.00% ||   0.01 %   || 1ms
1024.00KB || 28 || 1  || 14.29% ||   0.00 %   || 1ms
1024.00KB || 28 || 2  || 14.29% ||   0.00 %   || 1ms
1024.00KB || 28 || 3  || 14.29% ||   0.00 %   || 1ms
1024.00KB || 28 || 4  || 14.29% ||   0.00 %   || 1ms
1024.00KB || 28 || 5  || 14.29% ||   0.00 %   || 1ms
1024.00KB || 32 || 1  ||  0.00% ||   0.00 %   || 2ms
1024.00KB || 32 || 2  ||  0.00% ||   0.00 %   || 1ms
1024.00KB || 32 || 3  ||  0.00% ||   0.00 %   || 1ms
1024.00KB || 32 || 4  ||  0.00% ||   0.00 %   || 1ms
1024.00KB || 32 || 5  ||  0.00% ||   0.00 %   || 1ms
1024.00KB || 36 || 1  || 11.11% ||   0.01 %   || 1ms
1024.00KB || 36 || 2  || 11.11% ||   0.01 %   || 1ms
1024.00KB || 36 || 3  || 11.11% ||   0.01 %   || 1ms
1024.00KB || 36 || 4  || 11.11% ||   0.01 %   || 1ms
1024.00KB || 36 || 5  || 11.11% ||   0.01 %   || 1ms
1024.00KB || 40 || 1  ||  0.00% ||   0.01 %   || 1ms
1024.00KB || 40 || 2  ||  0.00% ||   0.01 %   || 1ms
1024.00KB || 40 || 3  ||  0.00% ||   0.01 %   || 1ms
1024.00KB || 40 || 4  ||  0.00% ||   0.01 %   || 1ms
1024.00KB || 40 || 5  ||  0.00% ||   0.01 %   || 1ms
1024.00KB || 44 || 1  ||  9.09% ||   0.01 %   || 1ms
1024.00KB || 44 || 2  ||  9.09% ||   0.01 %   || 1ms
1024.00KB || 44 || 3  ||  9.09% ||   0.01 %   || 1ms
1024.00KB || 44 || 4  ||  9.09% ||   0.01 %   || 1ms
1024.00KB || 44 || 5  ||  9.09% ||   0.01 %   || 1ms
1024.00KB || 48 || 1  ||  0.00% ||   0.01 %   || 1ms
1024.00KB || 48 || 2  ||  0.00% ||   0.01 %   || 0ms
1024.00KB || 48 || 3  ||  0.00% ||   0.01 %   || 1ms
1024.00KB || 48 || 4  ||  0.00% ||   0.01 %   || 1ms
1024.00KB || 48 || 5  ||  0.00% ||   0.01 %   || 1ms
1024.00KB || 52 || 1  ||  7.69% ||   0.01 %   || 1ms
1024.00KB || 52 || 2  ||  7.69% ||   0.01 %   || 1ms
1024.00KB || 52 || 3  ||  7.69% ||   0.01 %   || 1ms
1024.00KB || 52 || 4  ||  7.69% ||   0.01 %   || 0ms
1024.00KB || 52 || 5  ||  7.69% ||   0.01 %   || 1ms
1024.00KB || 56 || 1  ||  0.00% ||   0.01 %   || 1ms
1024.00KB || 56 || 2  ||  0.00% ||   0.01 %   || 1ms
1024.00KB || 56 || 3  ||  0.00% ||   0.01 %   || 1ms
1024.00KB || 56 || 4  ||  0.00% ||   0.01 %   || 0ms
1024.00KB || 56 || 5  ||  0.00% ||   0.01 %   || 1ms
1024.00KB || 60 || 1  ||  6.67% ||   0.01 %   || 1ms
1024.00KB || 60 || 2  ||  6.67% ||   0.01 %   || 1ms
1024.00KB || 60 || 3  ||  6.67% ||   0.01 %   || 0ms
1024.00KB || 60 || 4  ||  6.67% ||   0.01 %   || 1ms
1024.00KB || 60 || 5  ||  6.67% ||   0.01 %   || 1ms
1024.00KB || 64 || 1  ||  0.00% ||   0.01 %   || 1ms
1024.00KB || 64 || 2  ||  0.00% ||   0.01 %   || 0ms
1024.00KB || 64 || 3  ||  0.00% ||   0.01 %   || 1ms
1024.00KB || 64 || 4  ||  0.00% ||   0.01 %   || 1ms
1024.00KB || 64 || 5  ||  0.00% ||   0.01 %   || 1ms
   2.00MB || 28 || 1  || 14.29% ||   0.00 %   || 2ms
   2.00MB || 28 || 2  || 14.29% ||   0.00 %   || 2ms
   2.00MB || 28 || 3  || 14.29% ||   0.00 %   || 3ms
   2.00MB || 28 || 4  || 14.29% ||   0.00 %   || 2ms
   2.00MB || 28 || 5  || 14.29% ||   0.00 %   || 3ms
   2.00MB || 32 || 1  ||  0.00% ||   0.00 %   || 2ms
```

```
    2.00MB || 32 || 2  ||  0.00% ||   0.00 %   || 2ms
    2.00MB || 32 || 3  ||  0.00% ||   0.00 %   || 3ms
    2.00MB || 32 || 4  ||  0.00% ||   0.00 %   || 2ms
    2.00MB || 32 || 5  ||  0.00% ||   0.00 %   || 3ms
    2.00MB || 36 || 1  || 11.11% ||   0.00 %   || 2ms
    2.00MB || 36 || 2  || 11.11% ||   0.00 %   || 2ms
    2.00MB || 36 || 3  || 11.11% ||   0.00 %   || 2ms
    2.00MB || 36 || 4  || 11.11% ||   0.00 %   || 2ms
    2.00MB || 36 || 5  || 11.11% ||   0.00 %   || 2ms
    2.00MB || 40 || 1  ||  0.00% ||   0.00 %   || 2ms
    2.00MB || 40 || 2  ||  0.00% ||   0.00 %   || 2ms
    2.00MB || 40 || 3  ||  0.00% ||   0.00 %   || 2ms
    2.00MB || 40 || 4  ||  0.00% ||   0.00 %   || 2ms
    2.00MB || 40 || 5  ||  0.00% ||   0.00 %   || 2ms
    2.00MB || 44 || 1  ||  9.09% ||   0.00 %   || 1ms
    2.00MB || 44 || 2  ||  9.09% ||   0.00 %   || 2ms
    2.00MB || 44 || 3  ||  9.09% ||   0.00 %   || 2ms
    2.00MB || 44 || 4  ||  9.09% ||   0.00 %   || 2ms
    2.00MB || 44 || 5  ||  9.09% ||   0.00 %   || 1ms
    2.00MB || 48 || 1  ||  0.00% ||   0.00 %   || 2ms
    2.00MB || 48 || 2  ||  0.00% ||   0.00 %   || 2ms
    2.00MB || 48 || 3  ||  0.00% ||   0.00 %   || 2ms
    2.00MB || 48 || 4  ||  0.00% ||   0.00 %   || 1ms
    2.00MB || 48 || 5  ||  0.00% ||   0.00 %   || 3ms
    2.00MB || 52 || 1  ||  7.69% ||   0.00 %   || 1ms
    2.00MB || 52 || 2  ||  7.69% ||   0.00 %   || 2ms
    2.00MB || 52 || 3  ||  7.69% ||   0.00 %   || 2ms
    2.00MB || 52 || 4  ||  7.69% ||   0.00 %   || 1ms
    2.00MB || 52 || 5  ||  7.69% ||   0.00 %   || 2ms
    2.00MB || 56 || 1  ||  0.00% ||   0.00 %   || 1ms
    2.00MB || 56 || 2  ||  0.00% ||   0.00 %   || 2ms
    2.00MB || 56 || 3  ||  0.00% ||   0.00 %   || 2ms
    2.00MB || 56 || 4  ||  0.00% ||   0.00 %   || 1ms
    2.00MB || 56 || 5  ||  0.00% ||   0.00 %   || 2ms
    2.00MB || 60 || 1  ||  6.67% ||   0.00 %   || 1ms
    2.00MB || 60 || 2  ||  6.67% ||   0.00 %   || 2ms
    2.00MB || 60 || 3  ||  6.67% ||   0.00 %   || 1ms
    2.00MB || 60 || 4  ||  6.67% ||   0.00 %   || 2ms
    2.00MB || 60 || 5  ||  6.67% ||   0.00 %   || 1ms
    2.00MB || 64 || 1  ||  0.00% ||   0.00 %   || 2ms
    2.00MB || 64 || 2  ||  0.00% ||   0.00 %   || 1ms
    2.00MB || 64 || 3  ||  0.00% ||   0.00 %   || 2ms
    2.00MB || 64 || 4  ||  0.00% ||   0.00 %   || 1ms
    2.00MB || 64 || 5  ||  0.00% ||   0.00 %   || 2ms
    4.00MB || 28 || 1  || 14.29% ||   0.00 %   || 5ms
    4.00MB || 28 || 2  || 14.29% ||   0.00 %   || 5ms
    4.00MB || 28 || 3  || 14.29% ||   0.00 %   || 5ms
    4.00MB || 28 || 4  || 14.29% ||   0.00 %   || 5ms
    4.00MB || 28 || 5  || 14.29% ||   0.00 %   || 4ms
    4.00MB || 32 || 1  ||  0.00% ||   0.00 %   || 6ms
    4.00MB || 32 || 2  ||  0.00% ||   0.00 %   || 4ms
    4.00MB || 32 || 3  ||  0.00% ||   0.00 %   || 4ms
    4.00MB || 32 || 4  ||  0.00% ||   0.00 %   || 5ms
    4.00MB || 32 || 5  ||  0.00% ||   0.00 %   || 5ms
    4.00MB || 36 || 1  || 11.11% ||   0.00 %   || 4ms
    4.00MB || 36 || 2  || 11.11% ||   0.00 %   || 4ms
    4.00MB || 36 || 3  || 11.11% ||   0.00 %   || 5ms
    4.00MB || 36 || 4  || 11.11% ||   0.00 %   || 4ms
    4.00MB || 36 || 5  || 11.11% ||   0.00 %   || 3ms
    4.00MB || 40 || 1  ||  0.00% ||   0.00 %   || 4ms
    4.00MB || 40 || 2  ||  0.00% ||   0.00 %   || 5ms
    4.00MB || 40 || 3  ||  0.00% ||   0.00 %   || 4ms
    4.00MB || 40 || 4  ||  0.00% ||   0.00 %   || 4ms
    4.00MB || 40 || 5  ||  0.00% ||   0.00 %   || 4ms
    4.00MB || 44 || 1  ||  9.09% ||   0.00 %   || 3ms
```

```
   4.00MB || 44 || 2  ||  9.09% ||   0.00 %   || 4ms
   4.00MB || 44 || 3  ||  9.09% ||   0.00 %   || 4ms
   4.00MB || 44 || 4  ||  9.09% ||   0.00 %   || 3ms
   4.00MB || 44 || 5  ||  9.09% ||   0.00 %   || 4ms
   4.00MB || 48 || 1  ||  0.00% ||   0.00 %   || 3ms
   4.00MB || 48 || 2  ||  0.00% ||   0.00 %   || 4ms
   4.00MB || 48 || 3  ||  0.00% ||   0.00 %   || 3ms
   4.00MB || 48 || 4  ||  0.00% ||   0.00 %   || 4ms
   4.00MB || 48 || 5  ||  0.00% ||   0.00 %   || 3ms
   4.00MB || 52 || 1  ||  7.69% ||   0.00 %   || 4ms
   4.00MB || 52 || 2  ||  7.69% ||   0.00 %   || 3ms
   4.00MB || 52 || 3  ||  7.69% ||   0.00 %   || 3ms
   4.00MB || 52 || 4  ||  7.69% ||   0.00 %   || 3ms
   4.00MB || 52 || 5  ||  7.69% ||   0.00 %   || 4ms
   4.00MB || 56 || 1  ||  0.00% ||   0.00 %   || 3ms
   4.00MB || 56 || 2  ||  0.00% ||   0.00 %   || 4ms
   4.00MB || 56 || 3  ||  0.00% ||   0.00 %   || 3ms
   4.00MB || 56 || 4  ||  0.00% ||   0.00 %   || 3ms
   4.00MB || 56 || 5  ||  0.00% ||   0.00 %   || 3ms
   4.00MB || 60 || 1  ||  6.67% ||   0.00 %   || 3ms
   4.00MB || 60 || 2  ||  6.67% ||   0.00 %   || 3ms
   4.00MB || 60 || 3  ||  6.67% ||   0.00 %   || 3ms
   4.00MB || 60 || 4  ||  6.67% ||   0.00 %   || 3ms
   4.00MB || 60 || 5  ||  6.67% ||   0.00 %   || 3ms
   4.00MB || 64 || 1  ||  0.00% ||   0.00 %   || 4ms
   4.00MB || 64 || 2  ||  0.00% ||   0.00 %   || 3ms
   4.00MB || 64 || 3  ||  0.00% ||   0.00 %   || 3ms
   4.00MB || 64 || 4  ||  0.00% ||   0.00 %   || 3ms
   4.00MB || 64 || 5  ||  0.00% ||   0.00 %   || 3ms
   8.00MB || 28 || 1  || 14.29% ||   0.00 %   || 10ms
   8.00MB || 28 || 2  || 14.29% ||   0.00 %   || 9ms
   8.00MB || 28 || 3  || 14.29% ||   0.00 %   || 11ms
   8.00MB || 28 || 4  || 14.29% ||   0.00 %   || 12ms
   8.00MB || 28 || 5  || 14.29% ||   0.00 %   || 11ms
   8.00MB || 32 || 1  ||  0.00% ||   0.00 %   || 10ms
   8.00MB || 32 || 2  ||  0.00% ||   0.00 %   || 11ms
   8.00MB || 32 || 3  ||  0.00% ||   0.00 %   || 10ms
   8.00MB || 32 || 4  ||  0.00% ||   0.00 %   || 10ms
   8.00MB || 32 || 5  ||  0.00% ||   0.00 %   || 10ms
   8.00MB || 36 || 1  || 11.11% ||   0.00 %   || 9ms
   8.00MB || 36 || 2  || 11.11% ||   0.00 %   || 9ms
   8.00MB || 36 || 3  || 11.11% ||   0.00 %   || 9ms
   8.00MB || 36 || 4  || 11.11% ||   0.00 %   || 9ms
   8.00MB || 36 || 5  || 11.11% ||   0.00 %   || 9ms
   8.00MB || 40 || 1  ||  0.00% ||   0.00 %   || 8ms
   8.00MB || 40 || 2  ||  0.00% ||   0.00 %   || 9ms
   8.00MB || 40 || 3  ||  0.00% ||   0.00 %   || 8ms
   8.00MB || 40 || 4  ||  0.00% ||   0.00 %   || 11ms
   8.00MB || 40 || 5  ||  0.00% ||   0.00 %   || 9ms
   8.00MB || 44 || 1  ||  9.09% ||   0.00 %   || 8ms
   8.00MB || 44 || 2  ||  9.09% ||   0.00 %   || 7ms
   8.00MB || 44 || 3  ||  9.09% ||   0.00 %   || 8ms
   8.00MB || 44 || 4  ||  9.09% ||   0.00 %   || 7ms
   8.00MB || 44 || 5  ||  9.09% ||   0.00 %   || 8ms
   8.00MB || 48 || 1  ||  0.00% ||   0.00 %   || 7ms
   8.00MB || 48 || 2  ||  0.00% ||   0.00 %   || 8ms
   8.00MB || 48 || 3  ||  0.00% ||   0.00 %   || 7ms
   8.00MB || 48 || 4  ||  0.00% ||   0.00 %   || 8ms
   8.00MB || 48 || 5  ||  0.00% ||   0.00 %   || 7ms
   8.00MB || 52 || 1  ||  7.69% ||   0.00 %   || 7ms
   8.00MB || 52 || 2  ||  7.69% ||   0.00 %   || 7ms
   8.00MB || 52 || 3  ||  7.69% ||   0.00 %   || 7ms
   8.00MB || 52 || 4  ||  7.69% ||   0.00 %   || 7ms
   8.00MB || 52 || 5  ||  7.69% ||   0.00 %   || 7ms
   8.00MB || 56 || 1  ||  0.00% ||   0.00 %   || 6ms
```

```
   8.00MB || 56 || 2  ||  0.00% ||   0.00 %   || 7ms
   8.00MB || 56 || 3  ||  0.00% ||   0.00 %   || 7ms
   8.00MB || 56 || 4  ||  0.00% ||   0.00 %   || 7ms
   8.00MB || 56 || 5  ||  0.00% ||   0.00 %   || 7ms
   8.00MB || 60 || 1  ||  6.67% ||   0.00 %   || 7ms
   8.00MB || 60 || 2  ||  6.67% ||   0.00 %   || 7ms
   8.00MB || 60 || 3  ||  6.67% ||   0.00 %   || 6ms
   8.00MB || 60 || 4  ||  6.67% ||   0.00 %   || 6ms
   8.00MB || 60 || 5  ||  6.67% ||   0.00 %   || 7ms
   8.00MB || 64 || 1  ||  0.00% ||   0.00 %   || 6ms
   8.00MB || 64 || 2  ||  0.00% ||   0.00 %   || 6ms
   8.00MB || 64 || 3  ||  0.00% ||   0.00 %   || 6ms
   8.00MB || 64 || 4  ||  0.00% ||   0.00 %   || 6ms
   8.00MB || 64 || 5  ||  0.00% ||   0.00 %   || 6ms
  16.00MB || 28 || 1  || 14.29% ||   0.00 %   || 21ms
  16.00MB || 28 || 2  || 14.29% ||   0.00 %   || 20ms
  16.00MB || 28 || 3  || 14.29% ||   0.00 %   || 21ms
  16.00MB || 28 || 4  || 14.29% ||   0.00 %   || 20ms
  16.00MB || 28 || 5  || 14.29% ||   0.00 %   || 20ms
  16.00MB || 32 || 1  ||  0.00% ||   0.00 %   || 20ms
  16.00MB || 32 || 2  ||  0.00% ||   0.00 %   || 21ms
  16.00MB || 32 || 3  ||  0.00% ||   0.00 %   || 20ms
  16.00MB || 32 || 4  ||  0.00% ||   0.00 %   || 20ms
  16.00MB || 32 || 5  ||  0.00% ||   0.00 %   || 21ms
  16.00MB || 36 || 1  || 11.11% ||   0.00 %   || 17ms
  16.00MB || 36 || 2  || 11.11% ||   0.00 %   || 17ms
  16.00MB || 36 || 3  || 11.11% ||   0.00 %   || 18ms
  16.00MB || 36 || 4  || 11.11% ||   0.00 %   || 18ms
  16.00MB || 36 || 5  || 11.11% ||   0.00 %   || 17ms
  16.00MB || 40 || 1  ||  0.00% ||   0.00 %   || 18ms
  16.00MB || 40 || 2  ||  0.00% ||   0.00 %   || 17ms
  16.00MB || 40 || 3  ||  0.00% ||   0.00 %   || 18ms
  16.00MB || 40 || 4  ||  0.00% ||   0.00 %   || 17ms
  16.00MB || 40 || 5  ||  0.00% ||   0.00 %   || 17ms
  16.00MB || 44 || 1  ||  9.09% ||   0.00 %   || 16ms
  16.00MB || 44 || 2  ||  9.09% ||   0.00 %   || 15ms
  16.00MB || 44 || 3  ||  9.09% ||   0.00 %   || 15ms
  16.00MB || 44 || 4  ||  9.09% ||   0.00 %   || 16ms
  16.00MB || 44 || 5  ||  9.09% ||   0.00 %   || 15ms
  16.00MB || 48 || 1  ||  0.00% ||   0.00 %   || 16ms
  16.00MB || 48 || 2  ||  0.00% ||   0.00 %   || 15ms
  16.00MB || 48 || 3  ||  0.00% ||   0.00 %   || 15ms
  16.00MB || 48 || 4  ||  0.00% ||   0.00 %   || 16ms
  16.00MB || 48 || 5  ||  0.00% ||   0.00 %   || 16ms
  16.00MB || 52 || 1  ||  7.69% ||   0.00 %   || 14ms
  16.00MB || 52 || 2  ||  7.69% ||   0.00 %   || 14ms
  16.00MB || 52 || 3  ||  7.69% ||   0.00 %   || 14ms
  16.00MB || 52 || 4  ||  7.69% ||   0.00 %   || 14ms
  16.00MB || 52 || 5  ||  7.69% ||   0.00 %   || 15ms
  16.00MB || 56 || 1  ||  0.00% ||   0.00 %   || 14ms
  16.00MB || 56 || 2  ||  0.00% ||   0.00 %   || 15ms
  16.00MB || 56 || 3  ||  0.00% ||   0.00 %   || 14ms
  16.00MB || 56 || 4  ||  0.00% ||   0.00 %   || 14ms
  16.00MB || 56 || 5  ||  0.00% ||   0.00 %   || 14ms
  16.00MB || 60 || 1  ||  6.67% ||   0.00 %   || 13ms
  16.00MB || 60 || 2  ||  6.67% ||   0.00 %   || 13ms
  16.00MB || 60 || 3  ||  6.67% ||   0.00 %   || 13ms
  16.00MB || 60 || 4  ||  6.67% ||   0.00 %   || 13ms
  16.00MB || 60 || 5  ||  6.67% ||   0.00 %   || 12ms
  16.00MB || 64 || 1  ||  0.00% ||   0.00 %   || 13ms
  16.00MB || 64 || 2  ||  0.00% ||   0.00 %   || 13ms
  16.00MB || 64 || 3  ||  0.00% ||   0.00 %   || 15ms
  16.00MB || 64 || 4  ||  0.00% ||   0.00 %   || 13ms
  16.00MB || 64 || 5  ||  0.00% ||   0.00 %   || 14ms
  32.00MB || 28 || 1  || 14.29% ||   0.00 %   || 41ms
```

```
  32.00MB || 28 || 2  || 14.29% ||   0.00 %   || 41ms
  32.00MB || 28 || 3  || 14.29% ||   0.00 %   || 42ms
  32.00MB || 28 || 4  || 14.29% ||   0.00 %   || 42ms
  32.00MB || 28 || 5  || 14.29% ||   0.00 %   || 42ms
  32.00MB || 32 || 1  ||  0.00% ||   0.00 %   || 41ms
  32.00MB || 32 || 2  ||  0.00% ||   0.00 %   || 42ms
  32.00MB || 32 || 3  ||  0.00% ||   0.00 %   || 41ms
  32.00MB || 32 || 4  ||  0.00% ||   0.00 %   || 41ms
  32.00MB || 32 || 5  ||  0.00% ||   0.00 %   || 42ms
  32.00MB || 36 || 1  || 11.11% ||   0.00 %   || 35ms
  32.00MB || 36 || 2  || 11.11% ||   0.00 %   || 36ms
  32.00MB || 36 || 3  || 11.11% ||   0.00 %   || 36ms
  32.00MB || 36 || 4  || 11.11% ||   0.00 %   || 36ms
  32.00MB || 36 || 5  || 11.11% ||   0.00 %   || 36ms
  32.00MB || 40 || 1  ||  0.00% ||   0.00 %   || 35ms
  32.00MB || 40 || 2  ||  0.00% ||   0.00 %   || 35ms
  32.00MB || 40 || 3  ||  0.00% ||   0.00 %   || 35ms
  32.00MB || 40 || 4  ||  0.00% ||   0.00 %   || 35ms
  32.00MB || 40 || 5  ||  0.00% ||   0.00 %   || 36ms
  32.00MB || 44 || 1  ||  9.09% ||   0.00 %   || 31ms
  32.00MB || 44 || 2  ||  9.09% ||   0.00 %   || 32ms
  32.00MB || 44 || 3  ||  9.09% ||   0.00 %   || 31ms
  32.00MB || 44 || 4  ||  9.09% ||   0.00 %   || 31ms
  32.00MB || 44 || 5  ||  9.09% ||   0.00 %   || 32ms
  32.00MB || 48 || 1  ||  0.00% ||   0.00 %   || 31ms
  32.00MB || 48 || 2  ||  0.00% ||   0.00 %   || 34ms
  32.00MB || 48 || 3  ||  0.00% ||   0.00 %   || 31ms
  32.00MB || 48 || 4  ||  0.00% ||   0.00 %   || 31ms
  32.00MB || 48 || 5  ||  0.00% ||   0.00 %   || 31ms
  32.00MB || 52 || 1  ||  7.69% ||   0.00 %   || 29ms
  32.00MB || 52 || 2  ||  7.69% ||   0.00 %   || 30ms
  32.00MB || 52 || 3  ||  7.69% ||   0.00 %   || 29ms
  32.00MB || 52 || 4  ||  7.69% ||   0.00 %   || 29ms
  32.00MB || 52 || 5  ||  7.69% ||   0.00 %   || 29ms
  32.00MB || 56 || 1  ||  0.00% ||   0.00 %   || 29ms
  32.00MB || 56 || 2  ||  0.00% ||   0.00 %   || 28ms
  32.00MB || 56 || 3  ||  0.00% ||   0.00 %   || 29ms
  32.00MB || 56 || 4  ||  0.00% ||   0.00 %   || 29ms
  32.00MB || 56 || 5  ||  0.00% ||   0.00 %   || 28ms
  32.00MB || 64 || 1  ||  0.00% ||   0.00 %   || 27ms
  32.00MB || 64 || 2  ||  0.00% ||   0.00 %   || 27ms
  32.00MB || 64 || 3  ||  0.00% ||   0.00 %   || 27ms
  32.00MB || 64 || 4  ||  0.00% ||   0.00 %   || 26ms
  32.00MB || 64 || 5  ||  0.00% ||   0.00 %   || 27ms
  64.00MB || 28 || 1  || 14.29% ||   0.00 %   || 83ms
  64.00MB || 28 || 2  || 14.29% ||   0.00 %   || 83ms
  64.00MB || 28 || 3  || 14.29% ||   0.00 %   || 83ms
  64.00MB || 28 || 4  || 14.29% ||   0.00 %   || 85ms
  64.00MB || 28 || 5  || 14.29% ||   0.00 %   || 88ms
  64.00MB || 32 || 1  ||  0.00% ||   0.00 %   || 84ms
  64.00MB || 32 || 2  ||  0.00% ||   0.00 %   || 85ms
  64.00MB || 32 || 3  ||  0.00% ||   0.00 %   || 83ms
  64.00MB || 32 || 4  ||  0.00% ||   0.00 %   || 84ms
  64.00MB || 32 || 5  ||  0.00% ||   0.00 %   || 83ms
  64.00MB || 36 || 1  || 11.11% ||   0.00 %   || 71ms
  64.00MB || 36 || 2  || 11.11% ||   0.00 %   || 72ms
  64.00MB || 36 || 3  || 11.11% ||   0.00 %   || 72ms
  64.00MB || 36 || 4  || 11.11% ||   0.00 %   || 72ms
  64.00MB || 36 || 5  || 11.11% ||   0.00 %   || 72ms
  64.00MB || 40 || 1  ||  0.00% ||   0.00 %   || 71ms
  64.00MB || 40 || 2  ||  0.00% ||   0.00 %   || 73ms
  64.00MB || 40 || 3  ||  0.00% ||   0.00 %   || 72ms
  64.00MB || 40 || 4  ||  0.00% ||   0.00 %   || 72ms
  64.00MB || 40 || 5  ||  0.00% ||   0.00 %   || 73ms
  64.00MB || 44 || 1  ||  9.09% ||   0.00 %   || 64ms
```

```
  64.00MB || 44 || 2  ||  9.09% ||   0.00 %  || 62ms
  64.00MB || 44 || 3  ||  9.09% ||   0.00 %  || 65ms
  64.00MB || 44 || 4  ||  9.09% ||   0.00 %  || 64ms
  64.00MB || 44 || 5  ||  9.09% ||   0.00 %  || 64ms
  64.00MB || 48 || 1  ||  0.00% ||   0.00 %  || 64ms
  64.00MB || 48 || 2  ||  0.00% ||   0.00 %  || 64ms
  64.00MB || 48 || 3  ||  0.00% ||   0.00 %  || 63ms
  64.00MB || 48 || 4  ||  0.00% ||   0.00 %  || 64ms
  64.00MB || 48 || 5  ||  0.00% ||   0.00 %  || 64ms
  64.00MB || 52 || 1  ||  7.69% ||   0.00 %  || 59ms
  64.00MB || 52 || 2  ||  7.69% ||   0.00 %  || 58ms
  64.00MB || 52 || 3  ||  7.69% ||   0.00 %  || 61ms
  64.00MB || 52 || 4  ||  7.69% ||   0.00 %  || 58ms
  64.00MB || 52 || 5  ||  7.69% ||   0.00 %  || 60ms
  64.00MB || 56 || 1  ||  0.00% ||   0.00 %  || 59ms
  64.00MB || 56 || 2  ||  0.00% ||   0.00 %  || 59ms
  64.00MB || 56 || 3  ||  0.00% ||   0.00 %  || 58ms
  64.00MB || 56 || 4  ||  0.00% ||   0.00 %  || 59ms
  64.00MB || 56 || 5  ||  0.00% ||   0.00 %  || 57ms
  64.00MB || 60 || 1  ||  6.67% ||   0.00 %  || 55ms
  64.00MB || 60 || 2  ||  6.67% ||   0.00 %  || 54ms
  64.00MB || 60 || 3  ||  6.67% ||   0.00 %  || 55ms
  64.00MB || 60 || 4  ||  6.67% ||   0.00 %  || 55ms
  64.00MB || 60 || 5  ||  6.67% ||   0.00 %  || 54ms
  64.00MB || 64 || 1  ||  0.00% ||   0.00 %  || 54ms
  64.00MB || 64 || 2  ||  0.00% ||   0.00 %  || 54ms
  64.00MB || 64 || 3  ||  0.00% ||   0.00 %  || 55ms
  64.00MB || 64 || 4  ||  0.00% ||   0.00 %  || 54ms
  64.00MB || 64 || 5  ||  0.00% ||   0.00 %  || 56ms
 128.00MB || 28 || 1  || 14.29% ||   0.00 %  || 169ms
 128.00MB || 28 || 2  || 14.29% ||   0.00 %  || 167ms
 128.00MB || 28 || 3  || 14.29% ||   0.00 %  || 168ms
 128.00MB || 28 || 4  || 14.29% ||   0.00 %  || 167ms
 128.00MB || 28 || 5  || 14.29% ||   0.00 %  || 168ms
 128.00MB || 32 || 1  ||  0.00% ||   0.00 %  || 168ms
 128.00MB || 32 || 2  ||  0.00% ||   0.00 %  || 168ms
 128.00MB || 32 || 3  ||  0.00% ||   0.00 %  || 168ms
 128.00MB || 32 || 4  ||  0.00% ||   0.00 %  || 168ms
 128.00MB || 32 || 5  ||  0.00% ||   0.00 %  || 168ms
 128.00MB || 36 || 1  || 11.11% ||   0.00 %  || 144ms
 128.00MB || 36 || 2  || 11.11% ||   0.00 %  || 146ms
 128.00MB || 36 || 3  || 11.11% ||   0.00 %  || 147ms
 128.00MB || 36 || 4  || 11.11% ||   0.00 %  || 145ms
 128.00MB || 36 || 5  || 11.11% ||   0.00 %  || 145ms
 128.00MB || 40 || 1  ||  0.00% ||   0.00 %  || 144ms
 128.00MB || 40 || 2  ||  0.00% ||   0.00 %  || 146ms
 128.00MB || 40 || 3  ||  0.00% ||   0.00 %  || 144ms
 128.00MB || 40 || 4  ||  0.00% ||   0.00 %  || 144ms
 128.00MB || 40 || 5  ||  0.00% ||   0.00 %  || 148ms
 128.00MB || 44 || 1  ||  9.09% ||   0.00 %  || 130ms
 128.00MB || 44 || 2  ||  9.09% ||   0.00 %  || 130ms
 128.00MB || 44 || 3  ||  9.09% ||   0.00 %  || 130ms
 128.00MB || 44 || 4  ||  9.09% ||   0.00 %  || 129ms
 128.00MB || 44 || 5  ||  9.09% ||   0.00 %  || 130ms
 128.00MB || 48 || 1  ||  0.00% ||   0.00 %  || 130ms
 128.00MB || 48 || 2  ||  0.00% ||   0.00 %  || 130ms
 128.00MB || 48 || 3  ||  0.00% ||   0.00 %  || 130ms
 128.00MB || 48 || 4  ||  0.00% ||   0.00 %  || 129ms
 128.00MB || 48 || 5  ||  0.00% ||   0.00 %  || 130ms
 128.00MB || 52 || 1  ||  7.69% ||   0.00 %  || 119ms
 128.00MB || 52 || 2  ||  7.69% ||   0.00 %  || 118ms
 128.00MB || 52 || 3  ||  7.69% ||   0.00 %  || 119ms
 128.00MB || 52 || 4  ||  7.69% ||   0.00 %  || 120ms
 128.00MB || 52 || 5  ||  7.69% ||   0.00 %  || 120ms
 128.00MB || 56 || 1  ||  0.00% ||   0.00 %  || 119ms
```

```
 128.00MB || 56 || 2  ||  0.00% ||   0.00 %   || 121ms
 128.00MB || 56 || 3  ||  0.00% ||   0.00 %   || 119ms
 128.00MB || 56 || 4  ||  0.00% ||   0.00 %   || 119ms
 128.00MB || 56 || 5  ||  0.00% ||   0.00 %   || 120ms
 128.00MB || 60 || 1  ||  6.67% ||   0.00 %   || 110ms
 128.00MB || 60 || 2  ||  6.67% ||   0.00 %   || 110ms
 128.00MB || 60 || 3  ||  6.67% ||   0.00 %   || 111ms
 128.00MB || 60 || 4  ||  6.67% ||   0.00 %   || 112ms
 128.00MB || 60 || 5  ||  6.67% ||   0.00 %   || 110ms
 128.00MB || 64 || 1  ||  0.00% ||   0.00 %   || 111ms
 128.00MB || 64 || 2  ||  0.00% ||   0.00 %   || 115ms
 128.00MB || 64 || 3  ||  0.00% ||   0.00 %   || 111ms
 128.00MB || 64 || 4  ||  0.00% ||   0.00 %   || 110ms
 128.00MB || 64 || 5  ||  0.00% ||   0.00 %   || 111ms
 256.00MB || 28 || 1  || 14.29% ||   0.00 %   || 337ms
 256.00MB || 28 || 2  || 14.29% ||   0.00 %   || 335ms
 256.00MB || 28 || 3  || 14.29% ||   0.00 %   || 336ms
 256.00MB || 28 || 4  || 14.29% ||   0.00 %   || 338ms
 256.00MB || 28 || 5  || 14.29% ||   0.00 %   || 338ms
 256.00MB || 32 || 1  ||  0.00% ||   0.00 %   || 339ms
 256.00MB || 32 || 2  ||  0.00% ||   0.00 %   || 336ms
 256.00MB || 32 || 3  ||  0.00% ||   0.00 %   || 336ms
 256.00MB || 32 || 4  ||  0.00% ||   0.00 %   || 336ms
 256.00MB || 32 || 5  ||  0.00% ||   0.00 %   || 336ms
 256.00MB || 36 || 1  || 11.11% ||   0.00 %   || 292ms
 256.00MB || 36 || 2  || 11.11% ||   0.00 %   || 291ms
 256.00MB || 36 || 3  || 11.11% ||   0.00 %   || 291ms
 256.00MB || 36 || 4  || 11.11% ||   0.00 %   || 295ms
 256.00MB || 36 || 5  || 11.11% ||   0.00 %   || 290ms
 256.00MB || 40 || 1  ||  0.00% ||   0.00 %   || 289ms
 256.00MB || 40 || 2  ||  0.00% ||   0.00 %   || 292ms
 256.00MB || 40 || 3  ||  0.00% ||   0.00 %   || 290ms
 256.00MB || 40 || 4  ||  0.00% ||   0.00 %   || 289ms
 256.00MB || 40 || 5  ||  0.00% ||   0.00 %   || 293ms
 256.00MB || 44 || 1  ||  9.09% ||   0.00 %   || 309ms
 256.00MB || 44 || 2  ||  9.09% ||   0.00 %   || 279ms
 256.00MB || 44 || 3  ||  9.09% ||   0.00 %   || 314ms
 256.00MB || 44 || 4  ||  9.09% ||   0.00 %   || 335ms
 256.00MB || 44 || 5  ||  9.09% ||   0.00 %   || 280ms
 256.00MB || 48 || 1  ||  0.00% ||   0.00 %   || 264ms
 256.00MB || 48 || 2  ||  0.00% ||   0.00 %   || 269ms
 256.00MB || 48 || 3  ||  0.00% ||   0.00 %   || 269ms
 256.00MB || 48 || 4  ||  0.00% ||   0.00 %   || 266ms
 256.00MB || 48 || 5  ||  0.00% ||   0.00 %   || 266ms
 256.00MB || 52 || 1  ||  7.69% ||   0.00 %   || 249ms
 256.00MB || 52 || 2  ||  7.69% ||   0.00 %   || 245ms
 256.00MB || 52 || 3  ||  7.69% ||   0.00 %   || 246ms
 256.00MB || 52 || 4  ||  7.69% ||   0.00 %   || 244ms
 256.00MB || 52 || 5  ||  7.69% ||   0.00 %   || 241ms
 256.00MB || 56 || 1  ||  0.00% ||   0.00 %   || 238ms
 256.00MB || 56 || 2  ||  0.00% ||   0.00 %   || 237ms
 256.00MB || 56 || 3  ||  0.00% ||   0.00 %   || 239ms
 256.00MB || 56 || 4  ||  0.00% ||   0.00 %   || 241ms
 256.00MB || 56 || 5  ||  0.00% ||   0.00 %   || 238ms
 256.00MB || 60 || 1  ||  6.67% ||   0.00 %   || 222ms
 256.00MB || 60 || 2  ||  6.67% ||   0.00 %   || 220ms
 256.00MB || 60 || 3  ||  6.67% ||   0.00 %   || 223ms
 256.00MB || 60 || 4  ||  6.67% ||   0.00 %   || 225ms
 256.00MB || 60 || 5  ||  6.67% ||   0.00 %   || 224ms
 256.00MB || 64 || 1  ||  0.00% ||   0.00 %   || 222ms
 256.00MB || 64 || 2  ||  0.00% ||   0.00 %   || 221ms
 256.00MB || 64 || 3  ||  0.00% ||   0.00 %   || 222ms
 256.00MB || 64 || 4  ||  0.00% ||   0.00 %   || 222ms
 256.00MB || 64 || 5  ||  0.00% ||   0.00 %   || 221ms
```

APPENDIX

FILES FOR MEMORY ALLOCATOR

1. spHeap.c

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>

#include "spHeap.h"


/*
 * SIZE LIST USED AS BELOW:
 *
-------------------------------------------------------------
BN  c  bs op     power  B          KB         MB
-------------------------------------------------------------
0  1  2  power  2     4         0.00390625    3.8147E-06
1  3  2  power  1     6         0.005859375       5.72205E-06
2  1  2  power  3     8         0.0078125     7.62939E-06
3  3  2  power  2     12        0.01171875    1.14441E-05
4  1  2  power  4     16        0.015625      1.52588E-05
5  3  2  power  3     24        0.0234375     2.28882E-05
6  1  2  power  5     32        0.03125          3.05176E-05
7  3  2  power  4     48        0.046875      4.57764E-05
8  1  2  power  6     64        0.0625        6.10352E-05
9  3  2  power  5     96        0.09375           9.15527E-05
10 1  2  power  7     128           0.125         0.00012207
11 3  2  power  6     192           0.1875        0.000183105
12 1  2  power  8     256           0.25          0.000244141
13 3  2  power  7     384           0.375         0.000366211
14 1  2  power  9     512           0.5              0.000488281
15 3  2  power  8     768           0.75          0.000732422
16 1  2  power  10    1024      1             0.000976563
17 3  2  power  9     1536      1.5               0.001464844
18 1  2  power  11    2048      2             0.001953125
19 3  2  power  10    3072      3             0.002929688
20 1  2  power  12    4096      4             0.00390625
21 3  2  power  11    6144      6             0.005859375
22 1  2  power  13    8192      8             0.0078125
23 3  2  power  12    12288     12            0.01171875
24 1  2  power  14    16384     16            0.015625
25 3  2  power  13    24576     24            0.0234375
26 1  2  power  15    32768     32            0.03125
27 3  2  power  14    49152     48            0.046875
28 1  2  power  16    65536     64            0.0625
29 3  2  power  15    98304     96            0.09375
30 1  2  power  17    131072    128               0.125
31 3  2  power  16    196608    192               0.1875
32 1  2  power  18    262144    256               0.25
33 3  2  power  17    393216    384               0.375
34 1  2  power  19    524288    512               0.5
35 3  2  power  18    786432    768               0.75
36 1  2  power  20    1048576       1024          1
37 3  2  power  19    1572864       1536          1.5
38 1  2  power  21    2097152       2048          2
39 3  2  power  20    3145728       3072          3
40 1  2  power  22    4194304       4096          4
41 3  2  power  21    6291456       6144          6
42 1  2  power  23    8388608       8192          8
43 3  2  power  22    12582912   12288         12
44 1  2  power  24    16777216   16384         16
45 3  2  power  23    25165824   24576         24
46 1  2  power  25    33554432   32768         32
47 3  2  power  24    50331648   49152         48
```

```
48 1  2  power  26    67108864   65536        64
49 3  2  power  25    100663296  98304        96
50 1  2  power  27    134217728  131072       128
51 3  2  power  26    201326592  196608       192
52 1  2  power  28    268435456  262144       256
-----------------------------------------------------------



 There are 53 lines above. So our ASL can have 53 elements numbered 0 to 52.
 In general, if we allocate 2^n MB, we will have
 number of list elements = 37+2n elements
 in ASL list of size 4B and above.
 (4B element, plus two each for each power of 2^n Bytes, one for 1x2^n and one for 3x2^(n-
2))

 Note: Buckets of size MIN_ALLOCATABLE_BYTES and MIN_ALLOCATABLE_BYTES*3/2 cannot be split
*/

//Helper Functions Signatures
int isPowerOfTwo(int n);

int bucket_num(int memSizeRequired);

int get_bucket_size(int bucket_num);

int correctedSize(int memSizeinBytes);

spHeap *createEmptySPHeap();

spHeap *createMinSPHeap();

spHeap *initialize_memory_structure(int heapBytes);

void printMemBlock(memBlock *inputBlock);

void printMemBucket(memBucket *inputBucket);

BucketBlock *checkSpaceAvailableBucket(spHeap *inputHeap, int spaceRequired);

BucketBlock *checkSpaceAvailableInBucket(spHeap *inputHeap, int bucket_num);

void addBlockToTail(spHeap *inputHeap, int bucket_num, memBlock *memory_block);

void removeCurrentBlockFromSpaceList(spHeap *inputHeap, int bucket_num, memBlock
*currentBlock);

BucketBlock *split(spHeap *inputHeap, BucketBlock *bucketHavingSpace, int spaceRequired);

HeapStats *initialize_heap_stats(int heapSizeInBytes);

void update_allocation_stats(spHeap *inputHeap, int space_requested, int space_allocated);

void printStats(spHeap *inputHeap);

void freeMemoryRecursr(spHeap *inputHeap, BucketBlock *bucketFreed);

int freeHelp(spHeap *inputHeap, BucketBlock *bucketFreed, int buddy_bucket_num, void
*buddyAddr, int buddyLo,
             int combine_31_or_22);

//End Helper Functions Signatures

memBlock *createMemBlock(void *address, int kval, int tag, int type, memBlock *prev,
memBlock *next) {
```

```
    memBlock *out = calloc(1, sizeof(memBlock));
    out->kval = kval;
    out->tag = tag;
    out->type = type;
    out->mem_address = address;
    out->prev = prev;
    out->next = next;
    return out;
}

spHeap *initializeMemory(int heapBytes) {
    spHeap *heap = initialize_memory_structure(heapBytes);
    int last_bucket_num = heap->num_buckets - 1;
    heap->memBuckets[last_bucket_num].numMemBlocks = 1;
    heap->memBuckets[last_bucket_num].head = createMemBlock(NULL, last_bucket_num,
AVAILABLE, 0, NULL, NULL);
    heap->memBuckets[last_bucket_num].tail = heap->memBuckets[last_bucket_num].head;

    // * Above has been slightly modified from Shen's original paper for convenience.
    // * c*2^k can be represented in 2 ways. Therefore Paper Ambiguous. We store the actual
bucket number

    // * We also store the address rather than computing it each time. This is for speed
over memory efficiency

    int bucket_size_required = heap->memBuckets[last_bucket_num].bucketSizeinB;
    //we use malloc rather than calloc, as we are returning large storage size.
    // Cleaning all of them into 0's may take a long time.
    heap->baseAddress = malloc(bucket_size_required);
    heap->memBuckets[last_bucket_num].head->mem_address = heap->baseAddress;
    return heap;
}

void printHeap(spHeap *inputHeap) {
    printf(""
           "\n-------------------------------------------------------------------------------
----------------"
           "\nDETAILED STUDY OF HEAP THROUGH THE SP HEAP ALLOCATION"
           "\n-------------------------------------------------------------------------------
----------------\n");
    printf("Number of Buckets in Heap = %d\n", inputHeap->num_buckets);
    printf("Smallest Bucket Size in Heap = %d\n", inputHeap->smallestBucketSize);
    printf("Largest Bucket Size in Heap = %d\n", inputHeap->largestBucketSize);
    printf("\nNow Printing the Buckets");

    for (int i = 0; i < inputHeap->num_buckets; ++i) {
        printf(""
               "\n-----------------------------------------------------"
               "\nBucket No: %d\t", i);
        printMemBucket(&(inputHeap->memBuckets[i]));

    }
    printf("\n-----------------------------------------------------\n");
    printStats(inputHeap);

}

BucketBlock *allocateMemory(spHeap *inputHeap, int spaceRequired, int showErrors) {
    if (spaceRequired > inputHeap->largestBucketSize) {
        if (showErrors) {
            printf("The space you requested: %d is too big\n", spaceRequired);
        }
        return NULL;
    }
    if (spaceRequired < inputHeap->smallestBucketSize) {
```

```
        if (showErrors) {
            printf("The space you requested: %d is too small\n", spaceRequired);
        }
        return NULL;
    }
    BucketBlock *bucketHavingSpace = checkSpaceAvailableBucket(inputHeap, spaceRequired);
    if (!bucketHavingSpace) {
        if (showErrors) {
            printf("The space you requested: %d is not available. Sorry!\n",
spaceRequired);

            float percentageFull =
                    (float) inputHeap->stats->total_size_allocated / (float) inputHeap-
>stats->total_size_of_heap;
            printf("(%d,%.2f)", spaceRequired, percentageFull);
        }
        return NULL;
    }
    BucketBlock *exactBucket = split(inputHeap, bucketHavingSpace, spaceRequired);
    if (exactBucket->bucket_num == bucket_num(spaceRequired) || exactBucket->bucket_num <
3) {
        inputHeap->stats->num_allocs += 1;
        update_allocation_stats(inputHeap, spaceRequired, inputHeap-
>memBuckets[exactBucket->bucket_num].bucketSizeinB);
        exactBucket->block->tag = RESERVED;
        exactBucket->block->memRequest = spaceRequired;
        return exactBucket;
    }
    if (showErrors) {
        printf("Something went wrong.\n");
    }
    return NULL;
}


void freeMemory(spHeap *inputHeap, BucketBlock *bucketFreed) {
    inputHeap->stats->num_deallocs++;
    inputHeap->stats->total_size_allocated -= inputHeap->memBuckets[bucketFreed-
>bucket_num].bucketSizeinB;
    inputHeap->stats->total_size_requested -= bucketFreed->block->memRequest;
    freeMemoryRecursr(inputHeap, bucketFreed);
}

//Helper Functions Below


int bucket_num(int memSizeRequired) {
    int corrected_memsize = correctedSize(memSizeRequired);
    if (corrected_memsize < MIN_ALLOCATABLE_BYTES) {
        printf("The size required is too small\n");
        return -1;
    }
    if (corrected_memsize > MAX_HEAP_SIZE) {
        printf("The size required is too large\n");
        return -1;
    }
    int logSize = (int) floor(log2((int) corrected_memsize));
    int logMin = (int) log2((int) MIN_ALLOCATABLE_BYTES);
    if (isPowerOfTwo(corrected_memsize)) {
        return (logSize - logMin) * 2;
    } else {
        return (logSize - logMin) * 2 + 1;
    }
}
```

```
int get_bucket_size(int bucket_num) {
    int power_of_two = bucket_num / 2;
    if (bucket_num % 2 == 0) {
        int two_power_bucketByTwo = (int) pow(2, power_of_two);
        return 4 * two_power_bucketByTwo;
    } else {
        int two_power_bucketByTwo = (int) pow(2, power_of_two);
        return 6 * two_power_bucketByTwo;
    }
}

int correctedSize(int memSizeinBytes) {
    //first we decide how much to allocate
    if (memSizeinBytes > MAX_HEAP_SIZE) {
        printf("The memory size you have input is too large to fit into inputHeap");
        return -1;
    }

    int logSize = (int) ceil(log2((int) memSizeinBytes));
    int twoPowerLogSize = (int) pow(2, logSize);
    int output_size;
    if (twoPowerLogSize * 3 / 4 >= memSizeinBytes) {
        output_size = twoPowerLogSize * 3 / 4;
    } else {
        output_size = twoPowerLogSize;
    }
    return output_size;
}

spHeap *createEmptySPHeap() {
    spHeap *out = calloc(1, sizeof(spHeap));
    out->smallestBucketSize = 0;
    out->largestBucketSize = 0;
    out->num_buckets = 0;
    out->memBuckets = NULL;
    out->stats = NULL;
    return out;
}

spHeap *createMinSPHeap() {
    spHeap *out = calloc(1, sizeof(spHeap));
    out->smallestBucketSize = MIN_ALLOCATABLE_BYTES;
    out->largestBucketSize = MIN_ALLOCATABLE_BYTES;
    out->num_buckets = 1;
    out->stats = initialize_heap_stats(MIN_ALLOCATABLE_BYTES);
    out->memBuckets = calloc(1, sizeof(memBucket));
    out->memBuckets[0].bucketSizeinB = MIN_ALLOCATABLE_BYTES;
    out->memBuckets[0].numMemBlocks = 1;
    out->memBuckets[0].head = createMemBlock(0, 0, 0, 0, NULL, NULL);
    out->memBuckets[0].tail = out->memBuckets[0].head;
    out->baseAddress = calloc(1, MIN_ALLOCATABLE_BYTES);
    out->memBuckets[0].head->mem_address = out->baseAddress;
    return out;
}

BucketBlock *checkSpaceAvailableInBucket(spHeap *inputHeap, int bucket_num) {
    if (bucket_num < 0 || bucket_num >= inputHeap->num_buckets) {
        printf("Please check the bucket Number input\n");
        return NULL;
    }
    memBlock *memBlockRover = inputHeap->memBuckets[bucket_num].head;
    while (memBlockRover) {
        if (memBlockRover->tag == AVAILABLE) {
            BucketBlock *out = calloc(1, sizeof(BucketBlock));
            out->bucket_num = bucket_num;
```

```
            out->block = memBlockRover;
            return out;
        }
        memBlockRover = memBlockRover->next;
    }
    return NULL;
}

BucketBlock *checkSpaceAvailableBucket(spHeap *inputHeap, int spaceRequired) {
    if (spaceRequired < 0 || spaceRequired > MAX_HEAP_SIZE) {
        printf("Please check the space required:%d that you have input\n", spaceRequired);
        return NULL;
    }
    if (spaceRequired < inputHeap->smallestBucketSize) {
        printf("The space requested:%d is too low. Please request atleast %d bytes\n",
spaceRequired,
               inputHeap->smallestBucketSize);
        return NULL;
    }
    if (spaceRequired > inputHeap->largestBucketSize) {
        printf("The space requested:%d is too high. Please reinitialize a larger Heap\n",
spaceRequired);
        return NULL;
    }
    int bucketNum = bucket_num(spaceRequired);
    BucketBlock *spaceAvlBucket = NULL;
    for (int i = bucketNum; i < inputHeap->num_buckets && !spaceAvlBucket; ++i) {
        spaceAvlBucket = checkSpaceAvailableInBucket(inputHeap, i);
    }
    return spaceAvlBucket;
}

BucketBlock *split(spHeap *inputHeap, BucketBlock *bucketHavingSpace, int spaceRequired) {
    if (bucketHavingSpace->bucket_num == bucket_num(spaceRequired)) { return
bucketHavingSpace; }
    if (bucketHavingSpace->bucket_num < 3) { return bucketHavingSpace; }
    inputHeap->stats->splits += 1;

    memBlock *current_block = bucketHavingSpace->block;
    int bucket_num = bucketHavingSpace->bucket_num;
    unsigned int current_bucket_size = inputHeap->memBuckets[bucket_num].bucketSizeinB;


    if (bucket_num % 2 == 0) {
        memBlock *triple_two_power_n_minus2 = createMemBlock(current_block->mem_address,
current_block->kval - 1,
                                                             AVAILABLE, 3, NULL, NULL);
        addBlockToTail(inputHeap, current_block->kval - 1, triple_two_power_n_minus2);
        void *new_pointer = current_block->mem_address;
        //int address_shift = current_bucket_size >> 2 + current_bucket_size >> 1;
        unsigned int address_shift = current_bucket_size * 3 / 4;
        new_pointer = new_pointer + address_shift;
        memBlock *single_two_power_n_minus2 = createMemBlock(new_pointer, current_block-
>kval - 4, AVAILABLE, 3, NULL,
                                                             NULL);
        addBlockToTail(inputHeap, current_block->kval - 4, single_two_power_n_minus2);


        if (spaceRequired <= (inputHeap->memBuckets[current_block->kval -
4].bucketSizeinB)) {
            bucketHavingSpace->bucket_num = current_block->kval - 4;
            bucketHavingSpace->block = single_two_power_n_minus2;
        } else {
            bucketHavingSpace->bucket_num = current_block->kval - 1;
            bucketHavingSpace->block = triple_two_power_n_minus2;
```

```
        }
        removeCurrentBlockFromSpaceList(inputHeap, bucket_num, current_block);
        return split(inputHeap, bucketHavingSpace, spaceRequired);

    } else {
        memBlock *two_power_n_minus1 = createMemBlock(current_block->mem_address,
current_block->kval - 1, AVAILABLE, 2,
                                                      NULL, NULL);
        addBlockToTail(inputHeap, current_block->kval - 1, two_power_n_minus1);

        void *new_pointer = current_block->mem_address;
        unsigned int address_shift = (current_bucket_size << 1u) / 3;
        new_pointer = new_pointer + address_shift;
        memBlock *two_power_n_minus2 = createMemBlock(new_pointer, current_block->kval - 3,
AVAILABLE, 1, NULL, NULL);
        addBlockToTail(inputHeap, current_block->kval - 3, two_power_n_minus2);


        if (spaceRequired <= inputHeap->memBuckets[current_block->kval - 3].bucketSizeinB)
{
            bucketHavingSpace->bucket_num = current_block->kval - 3;
            bucketHavingSpace->block = two_power_n_minus2;
        } else {
            bucketHavingSpace->bucket_num = current_block->kval - 1;
            bucketHavingSpace->block = two_power_n_minus1;
        }
        removeCurrentBlockFromSpaceList(inputHeap, bucket_num, current_block);
        return split(inputHeap, bucketHavingSpace, spaceRequired);
    }


}

spHeap *initialize_memory_structure(int heapBytes) {
    if (heapBytes < MIN_ALLOCATABLE_BYTES) {
        printf("Error. Number of bytes requested is too low\n");
        return createEmptySPHeap();
    }
    if (heapBytes > MAX_HEAP_SIZE) {
        printf("Error. Number of bytes requested is too large\n");
        return createEmptySPHeap();
    }
    //error checks
    int heapSizeActual = correctedSize(heapBytes);
    int num_memory_buckets = bucket_num(heapSizeActual) + 1;

    spHeap *out = calloc(1, sizeof(spHeap));
    out->smallestBucketSize = 4;
    out->largestBucketSize = heapSizeActual;
    out->num_buckets = num_memory_buckets;
    out->stats = initialize_heap_stats(heapSizeActual);

    out->memBuckets = calloc(num_memory_buckets, sizeof(memBucket));
    for (int i = 0; i < num_memory_buckets; ++i) {
        out->memBuckets[i].numMemBlocks = 0;
        out->memBuckets[i].bucketSizeinB = get_bucket_size(i);
        out->memBuckets[i].head = NULL;
        out->memBuckets[i].tail = NULL;
    }
    return out;
}

void printMemBucket(memBucket *inputBucket) {
    printf("Bucket Size in Bytes = %d\n", inputBucket->bucketSizeinB);
    printf("Number of Memory Blocks in this bucket = %d", inputBucket->numMemBlocks);
```

```
    if (inputBucket->numMemBlocks > 0) {
        memBlock *memBlockRover = inputBucket->head;
        while (memBlockRover) {
            printf("\n-->\t");
            printMemBlock(memBlockRover);
            memBlockRover = memBlockRover->next;

        }
    }
}

void printMemBlock(memBlock *inputBlock) {
    printf("[");
    printf("addr = %p", inputBlock->mem_address);
    printf(", (");
    printf("kval = %d", inputBlock->kval);
    printf(",");
    printf("tag = %d", inputBlock->tag);
    printf(",");
    printf("type = %d", inputBlock->type);

    printf(")]");

}

HeapStats *initialize_heap_stats(int heapSizeInBytes) {
    HeapStats *out = calloc(1, sizeof(HeapStats));
    out->num_allocs = 0;
    out->num_deallocs = 0;
    out->recombines = 0;
    out->splits = 0;
    out->total_size_allocated = 0;
    out->total_size_requested = 0;
    out->total_size_of_heap = heapSizeInBytes;
    return out;
}

void update_allocation_stats(spHeap *inputHeap, int space_requested, int space_allocated) {
    inputHeap->stats->num_allocs += 1;
    inputHeap->stats->total_size_requested += space_requested;
    inputHeap->stats->total_size_allocated += space_allocated;
}

void printStats(spHeap *inputHeap) {
    float internal_fragmentation = 0;
    if (inputHeap->stats->total_size_requested > 0) {
        internal_fragmentation =
                (float) (inputHeap->stats->total_size_allocated - inputHeap->stats-
>total_size_requested) /
                (float) (inputHeap->stats->total_size_requested);
    }
    float percentageFull = 0;
    if (inputHeap->stats->total_size_of_heap > 0) {
        percentageFull = (float) inputHeap->stats->total_size_allocated / (float)
inputHeap->stats->total_size_of_heap;
    }

    printf(""
           "\n-----------------------------------------"
           "\nSTATISTICS FOR SPHEAP"
           "\nNumber of Allocation Requests: \t\t%d"
           "\nNumber of De-allocation Requests: \t%d"
           "\nNumber of Splits: \t\t\t\t%d"
           "\nNumber of Recombines: \t\t\t%d"
           "\nTotal Size of Heap: \t\t\t%d"
```

```
           "\nTotal Size Requested: \t\t\t\t%d"
           "\nTotal Size Allocated: \t\t\t\t%d"
           "\nInternal Fragmentation: \t\t\t%.2f%%"
           "\nPercentage Full: \t\t\t\t%.2f%%"
           "\n----------------------------------------\n",
           inputHeap->stats->num_allocs, inputHeap->stats->num_deallocs, inputHeap->stats-
>splits,
           inputHeap->stats->recombines,
           inputHeap->stats->total_size_of_heap, inputHeap->stats->total_size_requested,
           inputHeap->stats->total_size_allocated,
           internal_fragmentation * 100, percentageFull * 100);
}


// Now we have the recombine stuff

BucketBlock *findRecombineBuddy(spHeap *inputHeap, void *buddyAddr, int bucket_num) {
    memBlock *rover = inputHeap->memBuckets[bucket_num].head;
    while (rover) {
        if (rover->mem_address == buddyAddr) {
            BucketBlock *out = calloc(1, sizeof(BucketBlock));
            out->bucket_num = bucket_num;
            out->block = rover;
            return out;
        }
        rover = rover->next;
    }
    return NULL;
}

BucketBlock *combine_buddies31(spHeap *inputHeap, BucketBlock *bucketLow, BucketBlock
*bucketHi) {
    int lowSize = inputHeap->memBuckets[bucketLow->bucket_num].bucketSizeinB;

    if (bucketLow->block->mem_address + lowSize != bucketHi->block->mem_address) {
        printf(""
               "Are you sure these two are buddies? Please check again.\n"
               "Bucket Low Address: %p, Bucket Hi Address: %p, Bucket Lo Size: %d",
               bucketLow->block->mem_address, bucketHi->block->mem_address, lowSize);
        return NULL;
    }
    BucketBlock *out = calloc(1, sizeof(BucketBlock));
    out->bucket_num = bucketLow->bucket_num + 1;
    memBlock *newBlock = createMemBlock(bucketLow->block->mem_address, bucketLow->block-
>kval + 1, AVAILABLE, 2, NULL,
                                        NULL);
    out->block = newBlock;
    addBlockToTail(inputHeap, bucketLow->bucket_num + 1, newBlock);
    removeCurrentBlockFromSpaceList(inputHeap, bucketHi->bucket_num, bucketHi->block);
    removeCurrentBlockFromSpaceList(inputHeap, bucketLow->bucket_num, bucketLow->block);
    return out;
}

BucketBlock *combine_buddies22(spHeap *inputHeap, BucketBlock *bucketLow, BucketBlock
*bucketHi) {
    int lowSize = inputHeap->memBuckets[bucketLow->bucket_num].bucketSizeinB;

    if (bucketLow->block->mem_address + lowSize != bucketHi->block->mem_address) {
        printf(""
               "Are you sure these two are buddies? Please check again.\n"
               "Bucket Low Address: %p, Bucket Hi Address: %p, Bucket Lo Size: %d",
               bucketLow->block->mem_address, bucketHi->block->mem_address, lowSize);
        return NULL;
    }
    BucketBlock *out = calloc(1, sizeof(BucketBlock));
```

```
    out->bucket_num = bucketLow->bucket_num + 1;
    memBlock *newBlock = createMemBlock(bucketLow->block->mem_address, bucketLow->block-
>kval + 1, AVAILABLE, 3, NULL,
                                        NULL);
    out->block = newBlock;
    addBlockToTail(inputHeap, bucketLow->bucket_num + 1, newBlock);
    removeCurrentBlockFromSpaceList(inputHeap, bucketHi->bucket_num, bucketHi->block);
    removeCurrentBlockFromSpaceList(inputHeap, bucketLow->bucket_num, bucketLow->block);
    return out;
}


int freeHelp(spHeap *inputHeap, BucketBlock *bucketFreed, int buddy_bucket_num, void
*buddyAddr, int buddyLo,
             int combine_31_or_22) {
    BucketBlock *buddy = NULL;
    if (buddy_bucket_num < inputHeap->num_buckets && buddy_bucket_num >= 0) {
        buddy = findRecombineBuddy(inputHeap, buddyAddr, buddy_bucket_num);
    }
    if (buddy && buddy->block->tag == AVAILABLE) {
        inputHeap->stats->recombines += 1;
        BucketBlock *new_bucket_to_free = NULL;
        if (combine_31_or_22 == COMBINE31 && buddyLo == BUDDYLO) {
            new_bucket_to_free = combine_buddies31(inputHeap, buddy, bucketFreed);
        }
        if (combine_31_or_22 == COMBINE31 && buddyLo == BUDDYHI) {
            new_bucket_to_free = combine_buddies31(inputHeap, bucketFreed, buddy);
        }
        if (combine_31_or_22 == COMBINE22 && buddyLo == BUDDYLO) {
            new_bucket_to_free = combine_buddies22(inputHeap, buddy, bucketFreed);
        }
        if (combine_31_or_22 == COMBINE22 && buddyLo == BUDDYHI) {
            new_bucket_to_free = combine_buddies22(inputHeap, bucketFreed, buddy);
        }
        if (new_bucket_to_free) { freeMemoryRecursr(inputHeap, new_bucket_to_free); }
        return 1;
    }
    return 0;
}

void freeMemoryRecursr(spHeap *inputHeap, BucketBlock *bucketFreed) {
    bucketFreed->block->tag = AVAILABLE;
    int bucket_num = bucketFreed->bucket_num;
    memBlock *memFreed = bucketFreed->block;
    int block_size = inputHeap->memBuckets[bucket_num].bucketSizeinB;
    if (bucket_num % 2 == 0 && memFreed->type == 3) {
        void *buddyAddr = memFreed->mem_address - block_size * 3;
        freeHelp(inputHeap, bucketFreed, bucket_num + 3, buddyAddr, BUDDYLO, COMBINE31);
        return;
    } else if (bucket_num % 2 == 1 && memFreed->type == 3) {
        void *buddyAddr = memFreed->mem_address + block_size;
        freeHelp(inputHeap, bucketFreed, bucket_num - 3, buddyAddr, BUDDYHI, COMBINE31);

    } else if (bucket_num % 2 == 0 && (memFreed->type == 2 || memFreed->type == 1)) {
        void *buddyAddr = memFreed->mem_address + block_size;
        int freed = freeHelp(inputHeap, bucketFreed, bucket_num - 2, buddyAddr, BUDDYHI,
COMBINE22);
        if (!freed) {
            buddyAddr = memFreed->mem_address - block_size * 2;
            freed = freeHelp(inputHeap, bucketFreed, bucket_num + 2, buddyAddr, BUDDYLO,
COMBINE22);
            if (!freed) {
                buddyAddr = memFreed->mem_address - block_size * 3;
                freeHelp(inputHeap, bucketFreed, bucket_num + 3, buddyAddr, BUDDYLO,
COMBINE31);
```

```
            }
        }
        return;
    }
}


void removeCurrentBlockFromSpaceList(spHeap *inputHeap, int bucket_num, memBlock
*currentBlock) {
    inputHeap->memBuckets[bucket_num].numMemBlocks -= 1;
    if (inputHeap->memBuckets[bucket_num].head == currentBlock) {
        inputHeap->memBuckets[bucket_num].head = currentBlock->next;
    }
    if (inputHeap->memBuckets[bucket_num].tail == currentBlock) {
        inputHeap->memBuckets[bucket_num].tail = currentBlock->prev;
    }
    if (currentBlock->prev) { currentBlock->prev->next = currentBlock->next; }
    if (currentBlock->next) { currentBlock->next->prev = currentBlock->prev; }
    free(currentBlock);
}

void addBlockToTail(spHeap *inputHeap, int bucket_num, memBlock *memory_block) {
    memory_block->prev = inputHeap->memBuckets[bucket_num].tail;
    memory_block->next = NULL;
    if (inputHeap->memBuckets[bucket_num].tail) {
        inputHeap->memBuckets[bucket_num].tail->next = memory_block;
    } else {
        inputHeap->memBuckets[bucket_num].head = memory_block;
    }

    inputHeap->memBuckets[bucket_num].tail = memory_block;
    inputHeap->memBuckets[bucket_num].numMemBlocks += 1;
}

void freeHeap(spHeap *inputHeap) {
    free(inputHeap->stats);
    for (int i = 0; i < inputHeap->num_buckets; ++i) {
        memBlock *rover = inputHeap->memBuckets[i].head;
        while (rover) {
            memBlock *next = rover->next;
            free(rover);
            rover = next;
        }
    }
    free(inputHeap->baseAddress);
    free(inputHeap->memBuckets);
    free(inputHeap);

}
```

2. Helpers.c

```c
#include <stdio.h>
#include "helpers.h"
#include "math.h"

void printBin(int memSize) {
    if (memSize <= 1024) {
        printf("%8dB", memSize);
        return;
    }
    if (memSize <= 1024 * 1024) {
        printf("%8.2fKB", (float) memSize / 1024);
        return;
    }
    printf("%8.2fMB", (float) memSize / 1024 / 1024);
    return;
}

int two_power(int n) {
    if (n < 0 || n > 31) {
        printf("Please input valid No.\n");
        return 0;
    }
    return 1u << n;
}

unsigned int nextPowerOf2(unsigned int n) {
    unsigned count = 0;
// First n in the below condition
// is for the case where n is 0
    if (n && !(n & (n - 1)))
        return n;

    while (n != 0) {
        n >>= 1u;
        count += 1;
    }
    return 1u << count;
}

int next_multiple_of8(int n) {
    if (n % 8 == 0)
        return n;
    return ((n / 8) + 1) * 8;
}

int isPowerOfTwo(int n) {
    if (n == 0)
        return 0;
    return (ceil(log2(n)) == floor(log2(n)));
}
```

3. oneBin.c

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

#include "oneBin.h"


oneBin *ob_start_kenobi(int memSize, int oneBinSize) {
    int memSizeActual = nextPowerOf2((unsigned int) abs(memSize));
    if (memSize < MINBIN_SIZE) {
        memSizeActual = MINBIN_SIZE;
    } else if (memSizeActual > 536870912) {
        memSizeActual = 536870912;
    }

    int onebin_actual = next_multiple_of8(oneBinSize);
    if (oneBinSize < MINBIN_SIZE) { onebin_actual = MINBIN_SIZE; }
    if (oneBinSize > memSizeActual) { onebin_actual = memSizeActual; }

    oneBin *out = calloc(1, sizeof(oneBin));
    out->num_chunks = 1;
    out->firstFree = malloc(memSizeActual);
    out->total_size = memSizeActual;
    out->total_allocated = 0;
    out->base_address = out->firstFree;
    out->onebin_size = onebin_actual;

    memChunk *firstChunk = out->firstFree;
    firstChunk->nextChunk = NULL;
    firstChunk->prevChunk = NULL;
    firstChunk->size = memSizeActual;
    return out;
}

void set_address(void *mem_location, int size, memChunk *next, memChunk *prev) {
    memChunk *new_mem = mem_location;
    new_mem->size = size;
    new_mem->nextChunk = next;
    new_mem->prevChunk = prev;
}

void *ob_wan_memory(oneBin *ob_heap) {
    void *address_to_return = ob_heap->firstFree;
    memChunk *freeMem = ob_heap->firstFree;
    if (freeMem->size >= 2 * ob_heap->onebin_size) {
        void *new_address = ob_heap->firstFree + ob_heap->onebin_size;
        ob_heap->firstFree = new_address;
        set_address(new_address, freeMem->size - ob_heap->onebin_size, freeMem-
>nextChunk, freeMem->prevChunk);
        ob_heap->total_allocated += ob_heap->onebin_size;
    } else if (freeMem->nextChunk == NULL && freeMem->size < 2 * ob_heap-
>onebin_size) {
        printf("The Space you Allocated is full. Obi Cannot Wan");
```

```
        return NULL;
    } else if (freeMem->nextChunk && freeMem->size < 2 * ob_heap->onebin_size) {
        memChunk *nextMem = freeMem->nextChunk;
        set_address(nextMem, nextMem->size, nextMem->nextChunk, freeMem-
>prevChunk);
        ob_heap->firstFree = nextMem;
        ob_heap->total_allocated += ob_heap->onebin_size;
        ob_heap->num_chunks -= 1;
    }
    return address_to_return;
}

void merge_mem(oneBin *ob, void *obis_memory, void *next_free, void *prev_free) {
    memChunk *this_mem = obis_memory;
    memChunk *next_mem = next_free;
    memChunk *prev_mem = prev_free;

    void *this_address = this_mem;
    void *prev_address = prev_mem;
    if (this_address + this_mem->size == next_free && prev_address && (prev_address
+ prev_mem->size == this_address)) {
        prev_mem->size += this_mem->size + next_mem->size;
        prev_mem->nextChunk = next_mem->nextChunk;
        ob->num_chunks -= 2;
    } else if (this_address + this_mem->size == next_free) {
        this_mem->size += next_mem->size;
        this_mem->nextChunk = next_mem->nextChunk;
        ob->num_chunks -= 1;
    } else if (prev_address && (prev_address + prev_mem->size == this_address)) {
        prev_mem->size += this_mem->size;
        prev_mem->nextChunk = this_mem->nextChunk;
        ob->num_chunks -= 1;
    }
}

void ob_free_la_mem(oneBin *ob, void *obis_memory) {
    if (obis_memory < ob->base_address || obis_memory > ob->base_address + ob-
>total_size) {
        printf("Please check the memory location entered!\n");
        return;
    }
    if (obis_memory < ob->firstFree) {
        ob->total_allocated -= ob->onebin_size;
        memChunk *firstChunk = obis_memory;
        set_address(obis_memory, ob->onebin_size, ob->firstFree, NULL);
        memChunk *secondChunk = ob->firstFree;
        secondChunk->prevChunk = obis_memory;
        merge_mem(ob, obis_memory, ob->firstFree, NULL);
        ob->firstFree = obis_memory;
        ob->num_chunks += 1;
        return;
    } else {
        memChunk *rover = ob->firstFree;
        while (rover && ((void *) rover) < obis_memory) { rover = rover->nextChunk;
}
```

```
        if (!rover) {
            printf("That memory was never allocated! Error Error!\n");
            return;
        }
        memChunk *prevChunk = rover->prevChunk;
        prevChunk->nextChunk = obis_memory;
        rover->prevChunk = obis_memory;
        memChunk *thisChunk = obis_memory;
        thisChunk->prevChunk = prevChunk;
        thisChunk->nextChunk = rover;
        thisChunk->size = ob->onebin_size;
        merge_mem(ob, thisChunk, rover, prevChunk);
        ob->num_chunks += 1;
        return;

    }
}

void lightSaber(oneBin *ob) {

    printf("\n------------------------------------------------------------------------
-------------------------"
           "\nPrinting OB's Heap"
           "\nob->total_size=%d\tob->onebin_size=%d\t\tob->total_allocated=%d\tob-
>num_chunks=%d"
           "\nob->base_address=%p\tob->firstFree=%p",
           ob->total_size, ob->onebin_size, ob->total_allocated, ob->num_chunks,
ob->base_address, ob->firstFree);
    memChunk *rover = ob->firstFree;
    printf("\n");
    while (rover) {
        printf("-->rover->size=%d,Current address=%p,next Address=%p,prev
Address=%p\n",
               rover->size, rover, rover->nextChunk, rover->prevChunk);
        rover = rover->nextChunk;
    }
    printf("------------------------------------------------------------------------
----------------------\n");
}

void freeB(oneBin *ob) {
    free(ob->base_address);
    free(ob);
}
```

4. testSPHeap.c

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include <time.h>

#include "spHeap.h"
#include "spHeapTester.h"

int main() {
```

```
    printf("Hello, World!\nsizeof(int)=%d\n", sizeof(int));
    //doing some pointer arithmetic
    int a;
    printf("\na+0:%p", &a);
    printf("\na+1:%p", &a + 1);
    printf("\na+2:%p", &a + 2);

    void *b = &a;
    printf("\nb+0:%p", b + 0);
    printf("\nb+1:%p", b + 1);
    printf("\nb+2:%p", b + 2);

    for (int i = 0; i < 257; ++i) {
        int heapSizeInMB = i;
        int logHeapSize = (int) ceil(log2((int) heapSizeInMB));
        printf("heapSizeInMB= %d, logHeapSize=%d\n", heapSizeInMB, logHeapSize);
    }
    //Checking correctedSize function
    for (int i = 0; i < 257; ++i) {
        int correct_size = correctedSize(i);
        printf("memory_request= %d, correct_size=%d\n", i, correct_size);
    }
    //Checking bucket_num function
    for (int i = 0; i < 257; ++i) {
        int correct_size = correctedSize(i);
        int bucket_number = bucket_num(correct_size);
        printf("memory_request= %d, correct_size=%d, bucket_num = %d,\n", i,
correct_size, bucket_number);
    }
    //Checking get_bucket_size function
    for (int i = 0; i < 56; ++i) {
        int bucket_num = i;
        int bucket_sizeInB = get_bucket_size(i);
        printf("bucket_num= %d, bucket_sizeInB=%d\n", bucket_num, bucket_sizeInB);
    }
    //Checking Memory Initialization
    spHeap *heap1 = initializeMemory(10);
    printHeap(heap1);
    spHeap *heap2 = initializeMemory(100);
    printHeap(heap2);
    spHeap *heap3 = initializeMemory(1000);
    printHeap(heap3);
    // Below you can note how, though we requested only 100B, anything between 4B
and 128B is accepted as request.
    // This is because 100 is initialized to bucket 128B.
    for (int i = -10; i < 150; ++i) {
        BucketBlock *spaceAvlBucket = checkSpaceAvailableBucket(heap2, i);
        printf("Requested %d bytes. Space is ", i);
        if (!spaceAvlBucket) {
            printf("not available in Heap2\n");
            continue;
        }
        printf("available in bucket %d of Heap 2\n", spaceAvlBucket->bucket_num);
    }
    for (int i = -10; i < 20; ++i) {
```

```
        BucketBlock *some_mem = allocateMemory(heap2, i, 1);
        if (some_mem) {
            printf("\n\nRequested Memory Size = %d, obtained Pointer = %p", i,
some_mem->block->mem_address);
            printHeap(heap2);
        }
    }
    for (int i = 0; i < 40; ++i) {
        BucketBlock *some_mem = allocateMemory(heap3, i, 1);
        if (some_mem) {
            printf("\n\nRequested Memory Size = %d, obtained Pointer = %p", i,
some_mem->block->mem_address);
            printHeap(heap3);
        }
    }
    //Repeated Requests of Same Size
    spHeap *heap4 = initializeMemory(1000);
    printHeap(heap4);
    int num_Allocs = 45;
    BucketBlock **bucketsAllocated = calloc(num_Allocs, sizeof(BucketBlock *));
    for (int i = 0; i < num_Allocs; ++i) {
        bucketsAllocated[i] = allocateMemory(heap4, i, 1);
        if (bucketsAllocated[i]) {
            printf("\n\nRequested Memory Size = %d, obtained Pointer = %p", i,
bucketsAllocated[i]->block->mem_address);
            printHeap(heap4);
        }
    }
    //Free memory check
    for (int i = 0; i < num_Allocs; ++i) {
        if (bucketsAllocated[i]) {
            printf(""
                   "---------------------------------\n"
                   "The Block %d to be freed is as below\n", i);
            printMemBlock(bucketsAllocated[i]->block);
            printf("\n---------------------------------\n");
            freeMemory(heap4, bucketsAllocated[i]);
            printHeap(heap4);
        } else {
            printf("Sorry the bucket: %d has not been allocated\n", i);
        }
    }

    //check some inistializations
    spHeap *heap5 = initializeMemory(100);
    typedef struct somestruct {
        int a;
        float f;
    } someStruct;

    BucketBlock *somemem = allocateMemory(heap5, sizeof(someStruct), 1);
    someStruct *ss = somemem->block->mem_address;
    ss->a = 5;
    ss->f = 2.2f;
    printf("\nss->a=%d,ss->f = %.2f\n", ss->a, ss->f);
```

```
    BucketBlock *someothermem = allocateMemory(heap5, sizeof(someStruct), 1);
    someStruct *ss2 = someothermem->block->mem_address;
    ss2->a = 10;
    ss2->f = 4.4f;
    printf("\nss2->a=%d,ss2->f = %.2f\n", ss2->a, ss2->f);

    freeMemory(heap5, somemem);
    freeMemory(heap5, someothermem);
    printHeap(heap5);

    /*
     * The all important Experiment now
    */

    //First the bounds of the experiment
    int num_trials = 5;
    //size of memory in power of 2
    int minMemSize = 4;
    int maxMemSize = 28;
    //size of bin asked
    int binmin = 24;
    int binmax = 64;
    int binchange = 4;

    typedef struct experimental_structure {
        int a;
        int b;
        int c;
    } SPHeapTesterStruct;

    printf(""
           "\n-------------------------------------------------------------------------
---"
           "\n-------------------------------------------------------------------------
---"
           "\n\t\tStarting the experiment for spHeap"
           "\n-------------------------------------------------------------------------
---"
           "\n-------------------------------------------------------------------------
---");

    clock_t start, end;
    int num_experiments = 0, timeCount = 0;
    double TotIntFrag = 0, TotExtFrag = 0, TotMSPerMB = 0;


    printf("\n\tMemSize\t||\t
Bin\t||\tCnt\t||\tIntFrag\t||\tExtFrag\t\t||\tTimeTaken\n");
    for (int memPower = minMemSize; memPower <= maxMemSize; memPower += 1) {
        int binmin_actual = binmin, binmax_actual = binmax, binchange_actual =
binchange;
        if (two_power(memPower) >= 1024 * 1024) {
            binmin_actual = two_power(memPower) / 1024 / 1024 * binmin;
            binmax_actual = two_power(memPower) / 1024 / 1024 * binmax;
```

```
            binchange_actual = two_power(memPower) / 1024 / 1024 * binchange;
        }
        for (int binSize = binmin_actual;
             binSize <= binmax_actual && binSize <= two_power(memPower); binSize +=
binchange_actual) {
            for (int trial_no = 0; trial_no < num_trials; ++trial_no) {
                int memFull = 0;
                int memSize = two_power(memPower);
                start = clock();
                int actual_bin_size = next_multiple_of8(binSize);
                int actual_mem_size = nextPowerOf2(memSize);
                int num_bins = actual_mem_size / actual_bin_size - 1;


//printf("actual_mem_size=%d,actual_bin_size=%d,num_bins=%d\n",actual_mem_size,actu
al_bin_size,num_bins);

                //The structures
                SPHeapTesterStruct **adobe_locations = calloc(num_bins,
sizeof(SPHeapTesterStruct *));
                BucketBlock **bucket_locations = calloc(num_bins,
sizeof(BucketBlock *));
                //The memory Heap

                spHeap *spEx = initializeMemory(memSize);
                for (int i = 0; i < num_bins && !memFull; ++i) {
                    bucket_locations[i] = allocateMemory(spEx, binSize, 0);
                    if (!bucket_locations[i]) {
                        memFull = 1;
                        num_bins = i;
                        continue;
                    }
                    adobe_locations[i] = bucket_locations[i]->block->mem_address;

                }

                for (int i = 0; i < num_bins; ++i) {
                    adobe_locations[i]->a = 100 * i + 1;
                    adobe_locations[i]->b = 100 * i + 2;
                    adobe_locations[i]->c = 100 * i + 3;
                }

                float intFrag = (float) (spEx->stats->total_size_allocated - spEx-
>stats->total_size_requested) /
                                (float) spEx->stats->total_size_requested;
                float extFrag =
                        1.00 - (float) spEx->stats->total_size_allocated / (float)
spEx->stats->total_size_of_heap;

                //for (int i = 0; i < num_bins; ++i) {
                // printf("For bin %d,a= %d,b= %d,c=%d\n",
                // i,adobe_locations[i]->a,adobe_locations[i]-
>b,adobe_locations[i]->c);
                // }
                for (int i = 0; i < num_bins; ++i) {
```

```
                    freeMemory(spEx, bucket_locations[i]);
                }

                freeHeap(spEx);
                free(bucket_locations);
                free(adobe_locations);
                end = clock();
                printBin(memSize);
                if(isnan(intFrag) || intFrag<0||intFrag>1){
                    intFrag=0.0;
                }
                printf("\t||\t%5d\t||\t%3d\t||\t%5.2f%%\t||\t %5.2f
%%\t||\t%lims\n",
                        binSize, trial_no + 1, intFrag * 100, extFrag * 100, end -
start);
                num_experiments++;
                TotIntFrag += intFrag;
                TotExtFrag += extFrag;
                if (end - start != 0) {
                    TotMSPerMB += (float) (end - start) / (float) actual_mem_size *
(float) (1024 * 1024);
                    //printf("TotMSPerMB=%5.2f\n",(float)(end-
start)/(float)actual_mem_size*(float)(1024*1024));
                    timeCount++;
                }

            }
        }
    }
    printf("The Experiment Statistics are as follows\n");
    printf("The Average Internal Fragmentation was %6.2f%%\n", TotIntFrag * 100 /
(float) num_experiments);
    printf("The Average External Fragmentation was %6.2f%%\n", TotExtFrag * 100 /
(float) num_experiments);
    printf("The Average Time Taken in ms. per MB was %6.2f\n", TotMSPerMB / (float)
timeCount);
}
```

5. testOneBin.c

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

#include "oneBinTester.h"
#include "spHeapTester.h"

int main() {
    printf("Hello, World!");
    oneBin *ob = ob_start_kenobi(256, 24);
    int *num_space = ob_wan_memory(ob);
    for (int i = 0; i < 50; i += 5) {
        *num_space = i;
        printf("Digit now saved = %d\n", *num_space);
    }

    struct somestruct {
        int a;
        int b;
        int c;
    };
    lightSaber(ob);
    struct somestruct *some1 = ob_wan_memory(ob);
    printf("address allocated = %p. size of somestruct = %d\n", some1, (int)
sizeof(struct somestruct));
    some1->a = 20;
    some1->b = 25;
    some1->c = 30;
    printf("some1 address = %p\n", some1);
    lightSaber(ob);
    struct somestruct *some2 = ob_wan_memory(ob);
    printf("some2 address = %p\n", some2);
    lightSaber(ob);
    some2->a = 5;
    some2->b = 10;
    some2->c = 15;
    printf("The numbers saved by you are"
           "\tfirst a = %d,\tfirst b = %d,\tfirst c = %d,"
           "\tsecond a = %d,\tsecond b = %d\tsecond c = %d\n",
           some1->a, some1->b, some1->c, some2->a, some2->b, some2->c);

    printf("Now testing freeing\n\n\n");
    printf("\n\nsome1, Freed Memory: %p", some1);
    ob_free_la_mem(ob, some1);
    lightSaber(ob);
    printf("some2, Freed Memory: %p", some2);
    ob_free_la_mem(ob, some2);
    lightSaber(ob);

    printf("num_space, Freed Memory: %p", num_space);
    ob_free_la_mem(ob, num_space);
    lightSaber(ob);
```

```
    //Another series of allocates and deallocates (want to allocate a hole)
    struct somestruct *some3 = ob_wan_memory(ob);
    printf("some3 address = %p\n", some3);
    lightSaber(ob);
    struct somestruct *some4 = ob_wan_memory(ob);
    printf("some4 address = %p\n", some4);
    lightSaber(ob);
    struct somestruct *some5 = ob_wan_memory(ob);
    printf("some5 address = %p\n", some5);
    lightSaber(ob);
    ob_free_la_mem(ob, some4);
    printf("gave up some memory = %p\n", some4);
    lightSaber(ob);
    struct somestruct *some6 = ob_wan_memory(ob);
    printf("some6 address = %p\n", some6);
    lightSaber(ob);
    freeB(ob);
    //All Working


    /*
     * Doing the loopy loop to output everything!
     */
    //First the bounds of the experiment
    int num_trials = 5;
    //size of memory in power of 2
    int minMemSize = 5;
    int maxMemSize = 28;
    //size of bin asked
    int binmin = 28;
    int binmax = 64;

    typedef struct experimental_structure {
        int a;
        int b;
        int c;
    } Adobe;

    printf(""
           "\n-------------------------------------------------------------------------
---"
           "\n-------------------------------------------------------------------------
---"
           "\n\t\t\t\tStarting the experiment"
           "\n-------------------------------------------------------------------------
---"
           "\n-------------------------------------------------------------------------
---");

    clock_t start, end;
    double cpu_time_used;

printf("\n\tMemSize\t||\tBin\t||\tCnt\t||\tIntFrag\t||\tExtFrag\t||\tTimeTaken\n");
    for (int memPower = minMemSize; memPower <= maxMemSize; memPower += 1) {
```

```
        for (int binSize = binmin; binSize <= binmax && binSize <=
two_power(memPower); binSize += 4) {
            for (int trial_no = 0; trial_no < num_trials; ++trial_no) {
                int memSize = two_power(memPower);
                start = clock();
                int actual_bin_size = next_multiple_of8(binSize);
                int actual_mem_size = nextPowerOf2(memSize);
                int num_bins = actual_mem_size / actual_bin_size - 1;
                Adobe **adobe_locations = calloc(num_bins, sizeof(Adobe *));
                oneBin *Xiobi = ob_start_kenobi(memSize, binSize);
                for (int i = 0; i < num_bins; ++i) {
                    adobe_locations[i] = ob_wan_memory(Xiobi);
                }
                for (int i = 0; i < num_bins; ++i) {
                    adobe_locations[i]->a = 100 * i + 1;
                    adobe_locations[i]->b = 100 * i + 2;
                    adobe_locations[i]->c = 100 * i + 3;
                }
                float intFrag = (float) (actual_bin_size - binSize) / (float)
binSize;
                float extFrag = 1.00 - (float) Xiobi->total_allocated / (float)
Xiobi->total_size;

                //for (int i = 0; i < num_bins; ++i) {
                // printf("For bin %d,a= %d,b= %d,c=%d\n",
                // i,adobe_locations[i]->a,adobe_locations[i]-
>b,adobe_locations[i]->c);
                // }
                for (int i = 0; i < num_bins; ++i) {
                    ob_free_la_mem(Xiobi, adobe_locations[i]);
                }
                //lightSaber(Xiobi);


                freeB(Xiobi);
                free(adobe_locations);
                end = clock();
                printBin(memSize);
                printf("\t||\t%d\t||\t%d\t||\t%2.2f%%\t||\t %2.2f %%\t||\t%lims\n",
                       binSize, trial_no + 1, 100 * intFrag, 100 * extFrag, end -
start);
            }
        }
    }


}
```

6. memComparisons.c

```c
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <math.h>

#include "oneBinTester.h"
#include "spHeapTester.h"

int main() {
    {
        /*
 * The all important Experiment now
*/

        //First the bounds of the experiment
        int num_trials = 5;
        //size of memory in power of 2
        int minMemSize = 4;
        int maxMemSize = 28;
        //size of bin asked
        int binmin = 24;
        int binmax = 64;
        int binchange = 4;

        typedef struct experimental_structure {
            int a;
            int b;
            int c;
        } SPHeapTesterStruct;

        printf(""
               "\n-------------------------------------------------------------------------------"
               "\n-------------------------------------------------------------------------------"
               "\n\t\tStarting the experiment for spHeap"
               "\n-------------------------------------------------------------------------------"
               "\n-------------------------------------------------------------------------------");

        clock_t start, end;
        int num_experiments = 0, timeCount = 0;
        double TotIntFrag = 0, TotExtFrag = 0, TotMSPerMB = 0;


        printf("\n\tMemSize\t||\t
Bin\t||\tCnt\t||\tIntFrag\t||\tExtFrag\t\t||\tTimeTaken\n");
        for (int memPower = minMemSize; memPower <= maxMemSize; memPower += 1) {
            int binmin_actual = binmin, binmax_actual = binmax, binchange_actual =
binchange;
            if (two_power(memPower) >= 1024 * 1024) {
                binmin_actual = two_power(memPower) / 1024 / 1024 * binmin;
```

```
                binmax_actual = two_power(memPower) / 1024 / 1024 * binmax;
                binchange_actual = two_power(memPower) / 1024 / 1024 * binchange;
            }
            for (int binSize = binmin_actual;
                 binSize <= binmax_actual && binSize <= two_power(memPower);
binSize += binchange_actual) {
                for (int trial_no = 0; trial_no < num_trials; ++trial_no) {
                    int memFull = 0;
                    int memSize = two_power(memPower);
                    start = clock();
                    int actual_bin_size = next_multiple_of8(binSize);
                    int actual_mem_size = nextPowerOf2(memSize);
                    int num_bins = actual_mem_size / actual_bin_size - 1;


//printf("actual_mem_size=%d,actual_bin_size=%d,num_bins=%d\n",actual_mem_size,actu
al_bin_size,num_bins);

                    //The structures
                    SPHeapTesterStruct **adobe_locations = calloc(num_bins,
sizeof(SPHeapTesterStruct *));
                    BucketBlock **bucket_locations = calloc(num_bins,
sizeof(BucketBlock *));
                    //The memory Heap

                    spHeap *spEx = initializeMemory(memSize);
                    for (int i = 0; i < num_bins && !memFull; ++i) {
                        bucket_locations[i] = allocateMemory(spEx, binSize, 0);
                        if (!bucket_locations[i]) {
                            memFull = 1;
                            num_bins = i;
                            continue;
                        }
                        adobe_locations[i] = bucket_locations[i]->block-
>mem_address;

                    }

                    for (int i = 0; i < num_bins; ++i) {
                        adobe_locations[i]->a = 100 * i + 1;
                        adobe_locations[i]->b = 100 * i + 2;
                        adobe_locations[i]->c = 100 * i + 3;
                    }

                    float intFrag = (float) (spEx->stats->total_size_allocated -
spEx->stats->total_size_requested) /
                                    (float) spEx->stats->total_size_requested;
                    float extFrag =
                            1.00 - (float) spEx->stats->total_size_allocated /
(float) spEx->stats->total_size_of_heap;

                    //for (int i = 0; i < num_bins; ++i) {
                    // printf("For bin %d,a= %d,b= %d,c=%d\n",
                    // i,adobe_locations[i]->a,adobe_locations[i]-
>b,adobe_locations[i]->c);
```

```
                    // }
                    for (int i = 0; i < num_bins; ++i) {
                        freeMemory(spEx, bucket_locations[i]);
                    }

                    freeHeap(spEx);
                    free(bucket_locations);
                    free(adobe_locations);
                    end = clock();
                    printBin(memSize);
                    if (isnan(intFrag) || intFrag < 0 || intFrag > 1) {
                        intFrag = 0.0;
                    }
                    printf("\t||\t%5d\t||\t%3d\t||\t%5.2f%%\t||\t %5.2f
%%\t||\t%lims\n",
                           binSize, trial_no + 1, intFrag * 100, extFrag * 100, end
- start);
                    num_experiments++;
                    TotIntFrag += intFrag;
                    TotExtFrag += extFrag;
                    if (end - start != 0) {
                        TotMSPerMB += (float) (end - start) / (float)
actual_mem_size * (float) (1024 * 1024);
                        //printf("TotMSPerMB=%5.2f\n",(float)(end-
start)/(float)actual_mem_size*(float)(1024*1024));
                        timeCount++;
                    }

                }
            }
        }
        printf("The Experiment Statistics are as follows\n");
        printf("The Average Internal Fragmentation was %6.2f%%\n", TotIntFrag * 100
/ (float) num_experiments);
        printf("The Average External Fragmentation was %6.2f%%\n", TotExtFrag * 100
/ (float) num_experiments);
        printf("The Average Time Taken in ms. per MB was %6.2f\n", TotMSPerMB /
(float) timeCount);


    }
    {
        //First the bounds of the experiment
        int num_trials = 5;
        //size of memory in power of 2
        int minMemSize = 5;
        int maxMemSize = 28;
        //size of bin asked
        int binmin = 28;
        int binmax = 64;

        typedef struct experimental_structure {
            int a;
            int b;
            int c;
```

```
        } Adobe;

        printf(""
               "\n-------------------------------------------------------------------------
-------"
               "\n-------------------------------------------------------------------------
-------"
               "\n\t\t\t\tStarting the experiment for one bin"
               "\n-------------------------------------------------------------------------
-------"
               "\n-------------------------------------------------------------------------
-------");

        clock_t start, end;
        int num_experiments = 0, timeCount = 0;
        double TotIntFrag = 0, TotExtFrag = 0, TotMSPerMB = 0;



printf("\n\tMemSize\t||\tBin\t||\tCnt\t||\tIntFrag\t||\tExtFrag\t\t||\tTimeTaken\n"
);
        for (int memPower = minMemSize; memPower <= maxMemSize; memPower += 1) {
            for (int binSize = binmin; binSize <= binmax && binSize <=
two_power(memPower); binSize += 4) {
                for (int trial_no = 0; trial_no < num_trials; ++trial_no) {
                    int memSize = two_power(memPower);
                    start = clock();
                    int actual_bin_size = next_multiple_of8(binSize);
                    int actual_mem_size = nextPowerOf2(memSize);
                    int num_bins = actual_mem_size / actual_bin_size - 1;
                    Adobe **adobe_locations = calloc(num_bins, sizeof(Adobe *));
                    oneBin *Xiobi = ob_start_kenobi(memSize, binSize);
                    for (int i = 0; i < num_bins; ++i) {
                        adobe_locations[i] = ob_wan_memory(Xiobi);
                    }
                    for (int i = 0; i < num_bins; ++i) {
                        adobe_locations[i]->a = 100 * i + 1;
                        adobe_locations[i]->b = 100 * i + 2;
                        adobe_locations[i]->c = 100 * i + 3;
                    }
                    float intFrag = (float) (actual_bin_size - binSize) / (float)
binSize;
                    float extFrag = 1.00 - (float) Xiobi->total_allocated / (float)
Xiobi->total_size;

                    //for (int i = 0; i < num_bins; ++i) {
                    // printf("For bin %d,a= %d,b= %d,c=%d\n",
                    // i,adobe_locations[i]->a,adobe_locations[i]-
>b,adobe_locations[i]->c);
                    // }
                    for (int i = 0; i < num_bins; ++i) {
                        ob_free_la_mem(Xiobi, adobe_locations[i]);
                    }
                    //lightSaber(Xiobi);
```

```
                    freeB(Xiobi);
                    free(adobe_locations);
                    end = clock();
                    printBin(memSize);
                    printf("\t||\t%d\t||\t%d\t||\t%5.2f%%\t||\t %5.2f
%%\t||\t%lims\n",
                            binSize, trial_no + 1, intFrag * 100, extFrag * 100, end
- start);
                    num_experiments++;
                    TotIntFrag += intFrag;
                    TotExtFrag += extFrag;
                    if (end - start != 0) {
                        TotMSPerMB += (float) (end - start) / (float)
actual_mem_size * (float) (1024 * 1024);
                        //printf("TotMSPerMB=%5.2f\n",(float)(end-
start)/(float)actual_mem_size*(float)(1024*1024));
                        timeCount++;
                    }

                }
            }
        }
        printf("The Experiment Statistics are as follows\n");
        printf("The Average Internal Fragmentation was %6.2f%%\n", TotIntFrag * 100
/ (float) num_experiments);
        printf("The Average External Fragmentation was %6.2f%%\n", TotExtFrag * 100
/ (float) num_experiments);
        printf("The Average Time Taken in ms. per MB was %6.2f\n", TotMSPerMB /
(float) timeCount);
    }

}
```

Result for polynomial arithmetic

```
0000000000F0D040
0000000000F0D060
0000000000F0D080
0000000000F0D0A0
0000000000F0D0C0
0000000000F0D0E0
0000000000F0D100
0000000000F0D120
0000000000F0D140
0000000000F0D160
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Testing poly Ops
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

## 1. Testing Generation, printing of polynomials and adding of monomials to polynomials
Printing a random 10 degree polynomial with 0 terms
0
Printing a random 10 degree polynomial with 2 terms
34.36 x^6  +  5.75 x^2
Printing a random 10 degree polynomial with 4 terms
26.14 x^6  +  18.96 x^3  +  14.48 x^2  +  90.92
Printing a random 10 degree polynomial with 6 terms
```

```
90.92 x^6  +  6.26 x^4   +  42.65 x^3  +  18.96 x^2  +  45.06 x^1  +  69.76
Printing a random 10 degree polynomial with 8 terms
50.18 x^8  +  69.76 x^6  +  42.02 x^5  +  57.72 x^4  +  45.06 x^3  +  42.65 x^2  +  29.03 x^1
+  6.26
Printing a random 20 degree polynomial with 0 terms
0
Printing a random 20 degree polynomial with 2 terms
34.36 x^20 +  5.75 x^12
Printing a random 20 degree polynomial with 4 terms
26.14 x^20 +  14.48 x^12 +  18.96 x^11 +  90.92 x^9
Printing a random 20 degree polynomial with 6 terms
90.92 x^20 +  6.26 x^17  +  18.96 x^12 +  42.65 x^11 +  69.76 x^9  +  45.06
Printing a random 20 degree polynomial with 8 terms
69.76 x^20 +  57.72 x^17 +  42.65 x^12 +  45.06 x^11 +  6.26 x^9   +  50.18 x^8  +  42.02 x^5
+  29.03
Printing a random 30 degree polynomial with 0 terms
0
Printing a random 30 degree polynomial with 2 terms
5.75 x^12  +  34.36 x^9
Printing a random 30 degree polynomial with 4 terms
90.92 x^15 +  14.48 x^12 +  26.14 x^9  +  18.96 x^7
Printing a random 30 degree polynomial with 6 terms
6.26 x^19  +  69.76 x^15 +  18.96 x^12 +  90.92 x^9  +  45.06 x^8  +  42.65 x^7
Printing a random 30 degree polynomial with 8 terms
57.72 x^19 +  42.02 x^16 +  6.26 x^15  +  50.18 x^13 +  42.65 x^12 +  69.76 x^9  +  29.03 x^8
+  45.06 x^7
Printing a random 40 degree polynomial with 0 terms
0
Printing a random 40 degree polynomial with 2 terms
34.36 x^17 +  5.75 x^12
Printing a random 40 degree polynomial with 4 terms
18.96 x^21 +  26.14 x^17 +  14.48 x^12 +  90.92 x^6
Printing a random 40 degree polynomial with 6 terms
45.06 x^24 +  42.65 x^21 +  90.92 x^17 +  18.96 x^12 +  69.76 x^6  +  6.26 x^4
Printing a random 40 degree polynomial with 8 terms
29.03 x^24 +  50.18 x^23 +  45.06 x^21 +  42.02 x^20 +  69.76 x^17 +  42.65 x^12 +  6.26 x^6
+  57.72 x^4
Printing a random 50 degree polynomial with 0 terms
0
Printing a random 50 degree polynomial with 2 terms
5.75 x^42  +  34.36 x^35
Printing a random 50 degree polynomial with 4 terms
14.48 x^42 +  26.14 x^35 +  90.92 x^17 +  18.96 x^11
Printing a random 50 degree polynomial with 6 terms
45.06 x^48 +  6.26 x^44  +  18.96 x^42 +  90.92 x^35 +  69.76 x^17 +  42.65 x^11
Printing a random 50 degree polynomial with 8 terms
29.03 x^48 +  57.72 x^44 +  42.65 x^42 +  50.18 x^38 +  69.76 x^35 +  6.26 x^17  +  45.06 x^11
+  42.02 x^5
Printing a random 60 degree polynomial with 0 terms
0
Printing a random 60 degree polynomial with 2 terms
34.36 x^27 +  5.75 x^12
Printing a random 60 degree polynomial with 4 terms
90.92 x^28 +  26.14 x^27 +  14.48 x^12 +  18.96
Printing a random 60 degree polynomial with 6 terms
6.26 x^58  +  45.06 x^36 +  69.76 x^28 +  90.92 x^27 +  18.96 x^12 +  42.65
Printing a random 60 degree polynomial with 8 terms
57.72 x^58 +  42.02 x^52 +  50.18 x^38 +  29.03 x^36 +  6.26 x^28  +  69.76 x^27 +  42.65 x^12
+  45.06
Printing a random 70 degree polynomial with 0 terms
0
Printing a random 70 degree polynomial with 2 terms
34.36 x^65 +  5.75 x^22
Printing a random 70 degree polynomial with 4 terms
26.14 x^65 +  90.92 x^60 +  18.96 x^47 +  14.48 x^22
```

```
Printing a random 70 degree polynomial with 6 terms
90.92 x^65 +  69.76 x^60 +  45.06 x^54 +  42.65 x^47 +  6.26 x^42  +  18.96 x^22
Printing a random 70 degree polynomial with 8 terms
69.76 x^65 +  42.02 x^64 +  6.26 x^60  +  29.03 x^54 +  45.06 x^47 +  57.72 x^42 +  50.18 x^33
+  42.65 x^22
Printing a random 80 degree polynomial with 0 terms
0
Printing a random 80 degree polynomial with 2 terms
34.36 x^53 +  5.75 x^52
Printing a random 80 degree polynomial with 4 terms
26.14 x^53 +  14.48 x^52 +  90.92 x^20 +  18.96 x^11
Printing a random 80 degree polynomial with 6 terms
90.92 x^53 +  18.96 x^52 +  45.06 x^28 +  69.76 x^20 +  42.65 x^11 +  6.26 x^4
Printing a random 80 degree polynomial with 8 terms
42.02 x^66 +  69.76 x^53 +  42.65 x^52 +  50.18 x^38 +  29.03 x^28 +  6.26 x^20  +  45.06 x^11
+  57.72 x^4
Printing a random 90 degree polynomial with 0 terms
0
Printing a random 90 degree polynomial with 2 terms
34.36 x^55 +  5.75 x^12
Printing a random 90 degree polynomial with 4 terms
26.14 x^55 +  18.96 x^35 +  90.92 x^24 +  14.48 x^12
Printing a random 90 degree polynomial with 6 terms
90.92 x^55 +  42.65 x^35 +  6.26 x^25  +  69.76 x^24 +  18.96 x^12 +  45.06 x^10
Printing a random 90 degree polynomial with 8 terms
42.02 x^76 +  50.18 x^73 +  69.76 x^55 +  45.06 x^35 +  57.72 x^25 +  6.26 x^24  +  42.65 x^12
+  29.03 x^10

## 2. Testing copyPoly
Printing a random 10 degree polynomial with 0 terms
0
Printing Copy of above
0
Printing a random 10 degree polynomial with 2 terms
34.36 x^6  +  5.75 x^2
Printing Copy of above
34.36 x^6  +  5.75 x^2
Printing a random 10 degree polynomial with 4 terms
26.14 x^6  +  18.96 x^3  +  14.48 x^2  +  90.92
Printing Copy of above
26.14 x^6  +  18.96 x^3  +  14.48 x^2  +  90.92
Printing a random 10 degree polynomial with 6 terms
90.92 x^6  +  6.26 x^4   +  42.65 x^3  +  18.96 x^2  +  45.06 x^1  +  69.76
Printing Copy of above
90.92 x^6  +  6.26 x^4   +  42.65 x^3  +  18.96 x^2  +  45.06 x^1  +  69.76
Printing a random 10 degree polynomial with 8 terms
50.18 x^8  +  69.76 x^6  +  42.02 x^5  +  57.72 x^4  +  45.06 x^3  +  42.65 x^2  +  29.03 x^1
+  6.26
Printing Copy of above
50.18 x^8  +  69.76 x^6  +  42.02 x^5  +  57.72 x^4  +  45.06 x^3  +  42.65 x^2  +  29.03 x^1
+  6.26
Printing a random 20 degree polynomial with 0 terms
0
Printing Copy of above
0
Printing a random 20 degree polynomial with 2 terms
34.36 x^20 +  5.75 x^12
Printing Copy of above
34.36 x^20 +  5.75 x^12
Printing a random 20 degree polynomial with 4 terms
26.14 x^20 +  14.48 x^12 +  18.96 x^11 +  90.92 x^9
Printing Copy of above
26.14 x^20 +  14.48 x^12 +  18.96 x^11 +  90.92 x^9
Printing a random 20 degree polynomial with 6 terms
90.92 x^20 +  6.26 x^17  +  18.96 x^12 +  42.65 x^11 +  69.76 x^9  +  45.06
```

```
Printing Copy of above
90.92 x^20 +  6.26 x^17  +  18.96 x^12 +  42.65 x^11 +  69.76 x^9  +  45.06
Printing a random 20 degree polynomial with 8 terms
69.76 x^20 +  57.72 x^17 +  42.65 x^12 +  45.06 x^11 +  6.26 x^9   +  50.18 x^8  +  42.02 x^5
+  29.03
Printing Copy of above
69.76 x^20 +  57.72 x^17 +  42.65 x^12 +  45.06 x^11 +  6.26 x^9   +  50.18 x^8  +  42.02 x^5
+  29.03
Printing a random 30 degree polynomial with 0 terms
0
Printing Copy of above
0
Printing a random 30 degree polynomial with 2 terms
5.75 x^12  +  34.36 x^9
Printing Copy of above
5.75 x^12  +  34.36 x^9
Printing a random 30 degree polynomial with 4 terms
90.92 x^15 +  14.48 x^12 +  26.14 x^9  +  18.96 x^7
Printing Copy of above
90.92 x^15 +  14.48 x^12 +  26.14 x^9  +  18.96 x^7
Printing a random 30 degree polynomial with 6 terms
6.26 x^19  +  69.76 x^15 +  18.96 x^12 +  90.92 x^9  +  45.06 x^8  +  42.65 x^7
Printing Copy of above
6.26 x^19  +  69.76 x^15 +  18.96 x^12 +  90.92 x^9  +  45.06 x^8  +  42.65 x^7
Printing a random 30 degree polynomial with 8 terms
57.72 x^19 +  42.02 x^16 +  6.26 x^15  +  50.18 x^13 +  42.65 x^12 +  69.76 x^9  +  29.03 x^8
+  45.06 x^7
Printing Copy of above
57.72 x^19 +  42.02 x^16 +  6.26 x^15  +  50.18 x^13 +  42.65 x^12 +  69.76 x^9  +  29.03 x^8
+  45.06 x^7
Printing a random 40 degree polynomial with 0 terms
0
Printing Copy of above
0
Printing a random 40 degree polynomial with 2 terms
34.36 x^17 +  5.75 x^12
Printing Copy of above
34.36 x^17 +  5.75 x^12
Printing a random 40 degree polynomial with 4 terms
18.96 x^21 +  26.14 x^17 +  14.48 x^12 +  90.92 x^6
Printing Copy of above
18.96 x^21 +  26.14 x^17 +  14.48 x^12 +  90.92 x^6
Printing a random 40 degree polynomial with 6 terms
45.06 x^24 +  42.65 x^21 +  90.92 x^17 +  18.96 x^12 +  69.76 x^6  +  6.26 x^4
Printing Copy of above
45.06 x^24 +  42.65 x^21 +  90.92 x^17 +  18.96 x^12 +  69.76 x^6  +  6.26 x^4
Printing a random 40 degree polynomial with 8 terms
29.03 x^24 +  50.18 x^23 +  45.06 x^21 +  42.02 x^20 +  69.76 x^17 +  42.65 x^12 +  6.26 x^6
+  57.72 x^4
Printing Copy of above
29.03 x^24 +  50.18 x^23 +  45.06 x^21 +  42.02 x^20 +  69.76 x^17 +  42.65 x^12 +  6.26 x^6
+  57.72 x^4
Printing a random 50 degree polynomial with 0 terms
0
Printing Copy of above
0
Printing a random 50 degree polynomial with 2 terms
5.75 x^42  +  34.36 x^35
Printing Copy of above
5.75 x^42  +  34.36 x^35
Printing a random 50 degree polynomial with 4 terms
14.48 x^42 +  26.14 x^35 +  90.92 x^17 +  18.96 x^11
Printing Copy of above
14.48 x^42 +  26.14 x^35 +  90.92 x^17 +  18.96 x^11
Printing a random 50 degree polynomial with 6 terms
```

```
45.06 x^48 +  6.26 x^44  +  18.96 x^42 +  90.92 x^35 +  69.76 x^17 +  42.65 x^11
Printing Copy of above
45.06 x^48 +  6.26 x^44  +  18.96 x^42 +  90.92 x^35 +  69.76 x^17 +  42.65 x^11
Printing a random 50 degree polynomial with 8 terms
29.03 x^48 +  57.72 x^44 +  42.65 x^42 +  50.18 x^38 +  69.76 x^35 +  6.26 x^17  +  45.06 x^11
+  42.02 x^5
Printing Copy of above
29.03 x^48 +  57.72 x^44 +  42.65 x^42 +  50.18 x^38 +  69.76 x^35 +  6.26 x^17  +  45.06 x^11
+  42.02 x^5
Printing a random 60 degree polynomial with 0 terms
0
Printing Copy of above
0
Printing a random 60 degree polynomial with 2 terms
34.36 x^27 +  5.75 x^12
Printing Copy of above
34.36 x^27 +  5.75 x^12
Printing a random 60 degree polynomial with 4 terms
90.92 x^28 +  26.14 x^27 +  14.48 x^12 +  18.96
Printing Copy of above
90.92 x^28 +  26.14 x^27 +  14.48 x^12 +  18.96
Printing a random 60 degree polynomial with 6 terms
6.26 x^58  +  45.06 x^36 +  69.76 x^28 +  90.92 x^27 +  18.96 x^12 +  42.65
Printing Copy of above
6.26 x^58  +  45.06 x^36 +  69.76 x^28 +  90.92 x^27 +  18.96 x^12 +  42.65
Printing a random 60 degree polynomial with 8 terms
57.72 x^58 +  42.02 x^52 +  50.18 x^38 +  29.03 x^36 +  6.26 x^28  +  69.76 x^27 +  42.65 x^12
+  45.06
Printing Copy of above
57.72 x^58 +  42.02 x^52 +  50.18 x^38 +  29.03 x^36 +  6.26 x^28  +  69.76 x^27 +  42.65 x^12
+  45.06
Printing a random 70 degree polynomial with 0 terms
0
Printing Copy of above
0
Printing a random 70 degree polynomial with 2 terms
34.36 x^65 +  5.75 x^22
Printing Copy of above
34.36 x^65 +  5.75 x^22
Printing a random 70 degree polynomial with 4 terms
26.14 x^65 +  90.92 x^60 +  18.96 x^47 +  14.48 x^22
Printing Copy of above
26.14 x^65 +  90.92 x^60 +  18.96 x^47 +  14.48 x^22
Printing a random 70 degree polynomial with 6 terms
90.92 x^65 +  69.76 x^60 +  45.06 x^54 +  42.65 x^47 +  6.26 x^42  +  18.96 x^22
Printing Copy of above
90.92 x^65 +  69.76 x^60 +  45.06 x^54 +  42.65 x^47 +  6.26 x^42  +  18.96 x^22
Printing a random 70 degree polynomial with 8 terms
69.76 x^65 +  42.02 x^64 +  6.26 x^60  +  29.03 x^54 +  45.06 x^47 +  57.72 x^42 +  50.18 x^33
+  42.65 x^22
Printing Copy of above
69.76 x^65 +  42.02 x^64 +  6.26 x^60  +  29.03 x^54 +  45.06 x^47 +  57.72 x^42 +  50.18 x^33
+  42.65 x^22
Printing a random 80 degree polynomial with 0 terms
0
Printing Copy of above
0
Printing a random 80 degree polynomial with 2 terms
34.36 x^53 +  5.75 x^52
Printing Copy of above
34.36 x^53 +  5.75 x^52
Printing a random 80 degree polynomial with 4 terms
26.14 x^53 +  14.48 x^52 +  90.92 x^20 +  18.96 x^11
Printing Copy of above
26.14 x^53 +  14.48 x^52 +  90.92 x^20 +  18.96 x^11
```

```
Printing a random 80 degree polynomial with 6 terms
90.92 x^53 +  18.96 x^52 +  45.06 x^28 +  69.76 x^20 +  42.65 x^11 +  6.26 x^4
Printing Copy of above
90.92 x^53 +  18.96 x^52 +  45.06 x^28 +  69.76 x^20 +  42.65 x^11 +  6.26 x^4
Printing a random 80 degree polynomial with 8 terms
42.02 x^66 +  69.76 x^53 +  42.65 x^52 +  50.18 x^38 +  29.03 x^28 +  6.26 x^20  +  45.06 x^11
+  57.72 x^4
Printing Copy of above
42.02 x^66 +  69.76 x^53 +  42.65 x^52 +  50.18 x^38 +  29.03 x^28 +  6.26 x^20  +  45.06 x^11
+  57.72 x^4
Printing a random 90 degree polynomial with 0 terms
0
Printing Copy of above
0
Printing a random 90 degree polynomial with 2 terms
34.36 x^55 +  5.75 x^12
Printing Copy of above
34.36 x^55 +  5.75 x^12
Printing a random 90 degree polynomial with 4 terms
26.14 x^55 +  18.96 x^35 +  90.92 x^24 +  14.48 x^12
Printing Copy of above
26.14 x^55 +  18.96 x^35 +  90.92 x^24 +  14.48 x^12
Printing a random 90 degree polynomial with 6 terms
90.92 x^55 +  42.65 x^35 +  6.26 x^25  +  69.76 x^24 +  18.96 x^12 +  45.06 x^10
Printing Copy of above
90.92 x^55 +  42.65 x^35 +  6.26 x^25  +  69.76 x^24 +  18.96 x^12 +  45.06 x^10
Printing a random 90 degree polynomial with 8 terms
42.02 x^76 +  50.18 x^73 +  69.76 x^55 +  45.06 x^35 +  57.72 x^25 +  6.26 x^24  +  42.65 x^12
+  29.03 x^10
Printing Copy of above
42.02 x^76 +  50.18 x^73 +  69.76 x^55 +  45.06 x^35 +  57.72 x^25 +  6.26 x^24  +  42.65 x^12
+  29.03 x^10

## 3. Testing mult_monomial_toPoly
Printing a random 10 degree polynomial with 0 terms
0
Multiplying monomial above with coeff = 49.703365 and power = 1
0
Printing a random 10 degree polynomial with 2 terms
34.36 x^6  +  5.75 x^2
Multiplying monomial above with coeff = 26.137615 and power = 5
898.11 x^11    +  150.27 x^7
Printing a random 10 degree polynomial with 4 terms
26.14 x^6  +  18.96 x^3  +  14.48 x^2  +  90.92
Multiplying monomial above with coeff = 69.761467 and power = 6
1823.40 x^12   +  1322.91 x^9    +  1010.15 x^8    +  6342.96 x^6
Printing a random 10 degree polynomial with 6 terms
90.92 x^6  +  6.26 x^4   +  42.65 x^3  +  18.96 x^2  +  45.06 x^1  +  69.76
Multiplying monomial above with coeff = 57.715595 and power = 3
5247.71 x^9    +  361.12 x^7 +  2461.47 x^6    +  1094.48 x^5    +  2600.56 x^4    +  4026.32
x^3
Printing a random 10 degree polynomial with 8 terms
50.18 x^8  +  69.76 x^6  +  42.02 x^5  +  57.72 x^4  +  45.06 x^3  +  42.65 x^2  +  29.03 x^1
+  6.26
Multiplying monomial above with coeff = 68.327217 and power = 3
3428.90 x^11   +  4766.61 x^9    +  2871.00 x^8    +  3943.55 x^7    +  3078.70 x^6    +
2914.04 x^5    +  1983.58 x^4    +  427.52 x^3
Printing a random 20 degree polynomial with 0 terms
0
Multiplying monomial above with coeff = 49.703365 and power = 11
0
Printing a random 20 degree polynomial with 2 terms
34.36 x^20 +  5.75 x^12
Multiplying monomial above with coeff = 26.137615 and power = 15
898.11 x^35    +  150.27 x^27
```

```
Printing a random 20 degree polynomial with 4 terms
26.14 x^20 +  14.48 x^12 +  18.96 x^11 +  90.92 x^9
Multiplying monomial above with coeff = 69.761467 and power = 6
1823.40 x^26   +  1010.15 x^18   +  1322.91 x^17   +  6342.96 x^15
Printing a random 20 degree polynomial with 6 terms
90.92 x^20 +  6.26 x^17  +  18.96 x^12 +  42.65 x^11 +  69.76 x^9  +  45.06
Multiplying monomial above with coeff = 57.715595 and power = 13
5247.71 x^33   +  361.12 x^30    +  1094.48 x^25   +  2461.47 x^24   +  4026.32 x^22   +
2600.56 x^13
Printing a random 20 degree polynomial with 8 terms
69.76 x^20 +  57.72 x^17 +  42.65 x^12 +  45.06 x^11 +  6.26 x^9   +  50.18 x^8  +  42.02 x^5
+  29.03
Multiplying monomial above with coeff = 68.327217 and power = 3
4766.61 x^23   +  3943.55 x^20   +  2914.04 x^15   +  3078.70 x^14   +  427.52 x^12    +
3428.90 x^11   +  2871.00 x^8    +  1983.58 x^3
Printing a random 30 degree polynomial with 0 terms
0
Multiplying monomial above with coeff = 49.703365 and power = 11
0
Printing a random 30 degree polynomial with 2 terms
5.75 x^12  +  34.36 x^9
Multiplying monomial above with coeff = 26.137615 and power = 25
150.27 x^37    +  898.11 x^34
Printing a random 30 degree polynomial with 4 terms
90.92 x^15 +  14.48 x^12 +  26.14 x^9  +  18.96 x^7
Multiplying monomial above with coeff = 69.761467 and power = 26
6342.96 x^41   +  1010.15 x^38   +  1823.40 x^35   +  1322.91 x^33
Printing a random 30 degree polynomial with 6 terms
6.26 x^19  +  69.76 x^15 +  18.96 x^12 +  90.92 x^9  +  45.06 x^8  +  42.65 x^7
Multiplying monomial above with coeff = 57.715595 and power = 13
361.12 x^32    +  4026.32 x^28   +  1094.48 x^25   +  5247.71 x^22   +  2600.56 x^21   +
2461.47 x^20
Printing a random 30 degree polynomial with 8 terms
57.72 x^19 +  42.02 x^16 +  6.26 x^15  +  50.18 x^13 +  42.65 x^12 +  69.76 x^9  +  29.03 x^8
+  45.06 x^7
Multiplying monomial above with coeff = 68.327217 and power = 13
3943.55 x^32   +  2871.00 x^29   +  427.52 x^28    +  3428.90 x^26   +  2914.04 x^25   +
4766.61 x^22   +  1983.58 x^21   +  3078.70 x^20
Printing a random 40 degree polynomial with 0 terms
0
Multiplying monomial above with coeff = 49.703365 and power = 11
0
Printing a random 40 degree polynomial with 2 terms
34.36 x^17 +  5.75 x^12
Multiplying monomial above with coeff = 26.137615 and power = 15
898.11 x^32    +  150.27 x^27
Printing a random 40 degree polynomial with 4 terms
18.96 x^21 +  26.14 x^17 +  14.48 x^12 +  90.92 x^6
Multiplying monomial above with coeff = 69.761467 and power = 26
1322.91 x^47   +  1823.40 x^43   +  1010.15 x^38   +  6342.96 x^32
Printing a random 40 degree polynomial with 6 terms
45.06 x^24 +  42.65 x^21 +  90.92 x^17 +  18.96 x^12 +  69.76 x^6  +  6.26 x^4
Multiplying monomial above with coeff = 57.715595 and power = 13
2600.56 x^37   +  2461.47 x^34   +  5247.71 x^30   +  1094.48 x^25   +  4026.32 x^19   +
361.12 x^17
Printing a random 40 degree polynomial with 8 terms
29.03 x^24 +  50.18 x^23 +  45.06 x^21 +  42.02 x^20 +  69.76 x^17 +  42.65 x^12 +  6.26 x^6
+  57.72 x^4
Multiplying monomial above with coeff = 68.327217 and power = 3
1983.58 x^27   +  3428.90 x^26   +  3078.70 x^24   +  2871.00 x^23   +  4766.61 x^20   +
2914.04 x^15   +  427.52 x^9 +  3943.55 x^7
Printing a random 50 degree polynomial with 0 terms
0
Multiplying monomial above with coeff = 49.703365 and power = 41
0
```

```
Printing a random 50 degree polynomial with 2 terms
5.75 x^42  +  34.36 x^35
Multiplying monomial above with coeff = 26.137615 and power = 35
150.27 x^77    +  898.11 x^70
Printing a random 50 degree polynomial with 4 terms
14.48 x^42 +  26.14 x^35 +  90.92 x^17 +  18.96 x^11
Multiplying monomial above with coeff = 69.761467 and power = 46
1010.15 x^88   +  1823.40 x^81   +  6342.96 x^63   +  1322.91 x^57
Printing a random 50 degree polynomial with 6 terms
45.06 x^48 +  6.26 x^44  +  18.96 x^42 +  90.92 x^35 +  69.76 x^17 +  42.65 x^11
Multiplying monomial above with coeff = 57.715595 and power = 43
2600.56 x^91   +  361.12 x^87    +  1094.48 x^85   +  5247.71 x^78   +  4026.32 x^60   +
2461.47 x^54
Printing a random 50 degree polynomial with 8 terms
29.03 x^48 +  57.72 x^44 +  42.65 x^42 +  50.18 x^38 +  69.76 x^35 +  6.26 x^17  +  45.06 x^11
+  42.02 x^5
Multiplying monomial above with coeff = 68.327217 and power = 33
1983.58 x^81   +  3943.55 x^77   +  2914.04 x^75   +  3428.90 x^71   +  4766.61 x^68   +
427.52 x^50    +  3078.70 x^44   +  2871.00 x^38
Printing a random 60 degree polynomial with 0 terms
0
Multiplying monomial above with coeff = 49.703365 and power = 11
0
Printing a random 60 degree polynomial with 2 terms
34.36 x^27 +  5.75 x^12
Multiplying monomial above with coeff = 26.137615 and power = 55
898.11 x^82    +  150.27 x^67
Printing a random 60 degree polynomial with 4 terms
90.92 x^28 +  26.14 x^27 +  14.48 x^12 +  18.96
Multiplying monomial above with coeff = 69.761467 and power = 26
6342.96 x^54   +  1823.40 x^53   +  1010.15 x^38   +  1322.91 x^26
Printing a random 60 degree polynomial with 6 terms
6.26 x^58  +  45.06 x^36 +  69.76 x^28 +  90.92 x^27 +  18.96 x^12 +  42.65
Multiplying monomial above with coeff = 57.715595 and power = 13
361.12 x^71    +  2600.56 x^49   +  4026.32 x^41   +  5247.71 x^40   +  1094.48 x^25   +
2461.47 x^13
Printing a random 60 degree polynomial with 8 terms
57.72 x^58 +  42.02 x^52 +  50.18 x^38 +  29.03 x^36 +  6.26 x^28  +  69.76 x^27 +  42.65 x^12
+  45.06
Multiplying monomial above with coeff = 68.327217 and power = 43
3943.55 x^101  +  2871.00 x^95   +  3428.90 x^81   +  1983.58 x^79   +  427.52 x^71    +
4766.61 x^70   +  2914.04 x^55   +  3078.70 x^43
Printing a random 70 degree polynomial with 0 terms
0
Multiplying monomial above with coeff = 49.703365 and power = 21
0
Printing a random 70 degree polynomial with 2 terms
34.36 x^65 +  5.75 x^22
Multiplying monomial above with coeff = 26.137615 and power = 45
898.11 x^110   +  150.27 x^67
Printing a random 70 degree polynomial with 4 terms
26.14 x^65 +  90.92 x^60 +  18.96 x^47 +  14.48 x^22
Multiplying monomial above with coeff = 69.761467 and power = 16
1823.40 x^81   +  6342.96 x^76   +  1322.91 x^63   +  1010.15 x^38
Printing a random 70 degree polynomial with 6 terms
90.92 x^65 +  69.76 x^60 +  45.06 x^54 +  42.65 x^47 +  6.26 x^42  +  18.96 x^22
Multiplying monomial above with coeff = 57.715595 and power = 43
5247.71 x^108  +  4026.32 x^103  +  2600.56 x^97   +  2461.47 x^90   +  361.12 x^85    +
1094.48 x^65
Printing a random 70 degree polynomial with 8 terms
69.76 x^65 +  42.02 x^64 +  6.26 x^60  +  29.03 x^54 +  45.06 x^47 +  57.72 x^42 +  50.18 x^33
+  42.65 x^22
Multiplying monomial above with coeff = 68.327217 and power = 53
4766.61 x^118  +  2871.00 x^117  +  427.52 x^113   +  1983.58 x^107  +  3078.70 x^100  +
3943.55 x^95   +  3428.90 x^86   +  2914.04 x^75
```

```
Printing a random 80 degree polynomial with 0 terms
0
Multiplying monomial above with coeff = 49.703365 and power = 51
0
Printing a random 80 degree polynomial with 2 terms
34.36 x^53 +  5.75 x^52
Multiplying monomial above with coeff = 26.137615 and power = 15
898.11 x^68    +  150.27 x^67
Printing a random 80 degree polynomial with 4 terms
26.14 x^53 +  14.48 x^52 +  90.92 x^20 +  18.96 x^11
Multiplying monomial above with coeff = 69.761467 and power = 26
1823.40 x^79   +  1010.15 x^78   +  6342.96 x^46   +  1322.91 x^37
Printing a random 80 degree polynomial with 6 terms
90.92 x^53 +  18.96 x^52 +  45.06 x^28 +  69.76 x^20 +  42.65 x^11 +  6.26 x^4
Multiplying monomial above with coeff = 57.715595 and power = 53
5247.71 x^106  +  1094.48 x^105  +  2600.56 x^81   +  4026.32 x^73   +  2461.47 x^64   +
361.12 x^57
Printing a random 80 degree polynomial with 8 terms
42.02 x^66 +  69.76 x^53 +  42.65 x^52 +  50.18 x^38 +  29.03 x^28 +  6.26 x^20  +  45.06 x^11
+  57.72 x^4
Multiplying monomial above with coeff = 68.327217 and power = 3
2871.00 x^69   +  4766.61 x^56   +  2914.04 x^55   +  3428.90 x^41   +  1983.58 x^31   +
427.52 x^23    +  3078.70 x^14   +  3943.55 x^7
Printing a random 90 degree polynomial with 0 terms
0
Multiplying monomial above with coeff = 49.703365 and power = 11
0
Printing a random 90 degree polynomial with 2 terms
34.36 x^55 +  5.75 x^12
Multiplying monomial above with coeff = 26.137615 and power = 55
898.11 x^110   +  150.27 x^67
Printing a random 90 degree polynomial with 4 terms
26.14 x^55 +  18.96 x^35 +  90.92 x^24 +  14.48 x^12
Multiplying monomial above with coeff = 69.761467 and power = 86
1823.40 x^141  +  1322.91 x^121  +  6342.96 x^110  +  1010.15 x^98
Printing a random 90 degree polynomial with 6 terms
90.92 x^55 +  42.65 x^35 +  6.26 x^25  +  69.76 x^24 +  18.96 x^12 +  45.06 x^10
Multiplying monomial above with coeff = 57.715595 and power = 43
5247.71 x^98   +  2461.47 x^78   +  361.12 x^68    +  4026.32 x^67   +  1094.48 x^55   +
2600.56 x^53
Printing a random 90 degree polynomial with 8 terms
42.02 x^76 +  50.18 x^73 +  69.76 x^55 +  45.06 x^35 +  57.72 x^25 +  6.26 x^24  +  42.65 x^12
+  29.03 x^10
Multiplying monomial above with coeff = 68.327217 and power = 43
2871.00 x^119  +  3428.90 x^116  +  4766.61 x^98   +  3078.70 x^78   +  3943.55 x^68   +
427.52 x^67    +  2914.04 x^55   +  1983.58 x^53

## 4. Testing mult_monomial_toPoly_inplace
Printing a random 10 degree polynomial with 0 terms
0
Multiplying monomial in place above with coeff = 49.703365 and power = 1
0
Printing a random 10 degree polynomial with 2 terms
34.36 x^6  +  5.75 x^2
Multiplying monomial in place above with coeff = 26.137615 and power = 5
898.11 x^11    +  150.27 x^7
Printing a random 10 degree polynomial with 4 terms
26.14 x^6  +  18.96 x^3  +  14.48 x^2  +  90.92
Multiplying monomial in place above with coeff = 69.761467 and power = 6
1823.40 x^12   +  1322.91 x^9    +  1010.15 x^8    +  6342.96 x^6
Printing a random 10 degree polynomial with 6 terms
90.92 x^6  +  6.26 x^4   +  42.65 x^3  +  18.96 x^2  +  45.06 x^1  +  69.76
Multiplying monomial in place above with coeff = 57.715595 and power = 3
5247.71 x^9    +  361.12 x^7 +  2461.47 x^6    +  1094.48 x^5    +  2600.56 x^4    +  4026.32
x^3
```

```
Printing a random 10 degree polynomial with 8 terms
50.18 x^8  +  69.76 x^6  +  42.02 x^5  +  57.72 x^4  +  45.06 x^3  +  42.65 x^2  +  29.03 x^1
+  6.26
Multiplying monomial in place above with coeff = 68.327217 and power = 3
3428.90 x^11   +  4766.61 x^9    +  2871.00 x^8    +  3943.55 x^7    +  3078.70 x^6    +
2914.04 x^5    +  1983.58 x^4    +  427.52 x^3
Printing a random 20 degree polynomial with 0 terms
0
Multiplying monomial in place above with coeff = 49.703365 and power = 11
0
Printing a random 20 degree polynomial with 2 terms
34.36 x^20 +  5.75 x^12
Multiplying monomial in place above with coeff = 26.137615 and power = 15
898.11 x^35    +  150.27 x^27
Printing a random 20 degree polynomial with 4 terms
26.14 x^20 +  14.48 x^12 +  18.96 x^11 +  90.92 x^9
Multiplying monomial in place above with coeff = 69.761467 and power = 6
1823.40 x^26   +  1010.15 x^18   +  1322.91 x^17   +  6342.96 x^15
Printing a random 20 degree polynomial with 6 terms
90.92 x^20 +  6.26 x^17  +  18.96 x^12 +  42.65 x^11 +  69.76 x^9  +  45.06
Multiplying monomial in place above with coeff = 57.715595 and power = 13
5247.71 x^33   +  361.12 x^30    +  1094.48 x^25   +  2461.47 x^24   +  4026.32 x^22   +
2600.56 x^13
Printing a random 20 degree polynomial with 8 terms
69.76 x^20 +  57.72 x^17 +  42.65 x^12 +  45.06 x^11 +  6.26 x^9   +  50.18 x^8  +  42.02 x^5
+  29.03
Multiplying monomial in place above with coeff = 68.327217 and power = 3
4766.61 x^23   +  3943.55 x^20   +  2914.04 x^15   +  3078.70 x^14   +  427.52 x^12    +
3428.90 x^11   +  2871.00 x^8    +  1983.58 x^3
Printing a random 30 degree polynomial with 0 terms
0
Multiplying monomial in place above with coeff = 49.703365 and power = 11
0
Printing a random 30 degree polynomial with 2 terms
5.75 x^12  +  34.36 x^9
Multiplying monomial in place above with coeff = 26.137615 and power = 25
150.27 x^37    +  898.11 x^34
Printing a random 30 degree polynomial with 4 terms
90.92 x^15 +  14.48 x^12 +  26.14 x^9  +  18.96 x^7
Multiplying monomial in place above with coeff = 69.761467 and power = 26
6342.96 x^41   +  1010.15 x^38   +  1823.40 x^35   +  1322.91 x^33
Printing a random 30 degree polynomial with 6 terms
6.26 x^19  +  69.76 x^15 +  18.96 x^12 +  90.92 x^9  +  45.06 x^8  +  42.65 x^7
Multiplying monomial in place above with coeff = 57.715595 and power = 13
361.12 x^32    +  4026.32 x^28   +  1094.48 x^25   +  5247.71 x^22   +  2600.56 x^21   +
2461.47 x^20
Printing a random 30 degree polynomial with 8 terms
57.72 x^19 +  42.02 x^16 +  6.26 x^15  +  50.18 x^13 +  42.65 x^12 +  69.76 x^9  +  29.03 x^8
+  45.06 x^7
Multiplying monomial in place above with coeff = 68.327217 and power = 13
3943.55 x^32   +  2871.00 x^29   +  427.52 x^28    +  3428.90 x^26   +  2914.04 x^25   +
4766.61 x^22   +  1983.58 x^21   +  3078.70 x^20
Printing a random 40 degree polynomial with 0 terms
0
Multiplying monomial in place above with coeff = 49.703365 and power = 11
0
Printing a random 40 degree polynomial with 2 terms
34.36 x^17 +  5.75 x^12
Multiplying monomial in place above with coeff = 26.137615 and power = 15
898.11 x^32    +  150.27 x^27
Printing a random 40 degree polynomial with 4 terms
18.96 x^21 +  26.14 x^17 +  14.48 x^12 +  90.92 x^6
Multiplying monomial in place above with coeff = 69.761467 and power = 26
1322.91 x^47   +  1823.40 x^43   +  1010.15 x^38   +  6342.96 x^32
Printing a random 40 degree polynomial with 6 terms
```

```
45.06 x^24 +  42.65 x^21 +  90.92 x^17 +  18.96 x^12 +  69.76 x^6  +  6.26 x^4
Multiplying monomial in place above with coeff = 57.715595 and power = 13
2600.56 x^37   +  2461.47 x^34   +  5247.71 x^30   +  1094.48 x^25   +  4026.32 x^19   +
361.12 x^17
Printing a random 40 degree polynomial with 8 terms
29.03 x^24 +  50.18 x^23 +  45.06 x^21 +  42.02 x^20 +  69.76 x^17 +  42.65 x^12 +  6.26 x^6
+  57.72 x^4
Multiplying monomial in place above with coeff = 68.327217 and power = 3
1983.58 x^27   +  3428.90 x^26   +  3078.70 x^24   +  2871.00 x^23   +  4766.61 x^20   +
2914.04 x^15   +  427.52 x^9 +  3943.55 x^7
Printing a random 50 degree polynomial with 0 terms
0
Multiplying monomial in place above with coeff = 49.703365 and power = 41
0
Printing a random 50 degree polynomial with 2 terms
5.75 x^42  +  34.36 x^35
Multiplying monomial in place above with coeff = 26.137615 and power = 35
150.27 x^77    +  898.11 x^70
Printing a random 50 degree polynomial with 4 terms
14.48 x^42 +  26.14 x^35 +  90.92 x^17 +  18.96 x^11
Multiplying monomial in place above with coeff = 69.761467 and power = 46
1010.15 x^88   +  1823.40 x^81   +  6342.96 x^63   +  1322.91 x^57
Printing a random 50 degree polynomial with 6 terms
45.06 x^48 +  6.26 x^44  +  18.96 x^42 +  90.92 x^35 +  69.76 x^17 +  42.65 x^11
Multiplying monomial in place above with coeff = 57.715595 and power = 43
2600.56 x^91   +  361.12 x^87    +  1094.48 x^85   +  5247.71 x^78   +  4026.32 x^60   +
2461.47 x^54
Printing a random 50 degree polynomial with 8 terms
29.03 x^48 +  57.72 x^44 +  42.65 x^42 +  50.18 x^38 +  69.76 x^35 +  6.26 x^17  +  45.06 x^11
+  42.02 x^5
Multiplying monomial in place above with coeff = 68.327217 and power = 33
1983.58 x^81   +  3943.55 x^77   +  2914.04 x^75   +  3428.90 x^71   +  4766.61 x^68   +
427.52 x^50    +  3078.70 x^44   +  2871.00 x^38
Printing a random 60 degree polynomial with 0 terms
0
Multiplying monomial in place above with coeff = 49.703365 and power = 11
0
Printing a random 60 degree polynomial with 2 terms
34.36 x^27 +  5.75 x^12
Multiplying monomial in place above with coeff = 26.137615 and power = 55
898.11 x^82    +  150.27 x^67
Printing a random 60 degree polynomial with 4 terms
90.92 x^28 +  26.14 x^27 +  14.48 x^12 +  18.96
Multiplying monomial in place above with coeff = 69.761467 and power = 26
6342.96 x^54   +  1823.40 x^53   +  1010.15 x^38   +  1322.91 x^26
Printing a random 60 degree polynomial with 6 terms
6.26 x^58  +  45.06 x^36 +  69.76 x^28 +  90.92 x^27 +  18.96 x^12 +  42.65
Multiplying monomial in place above with coeff = 57.715595 and power = 13
361.12 x^71    +  2600.56 x^49   +  4026.32 x^41   +  5247.71 x^40   +  1094.48 x^25   +
2461.47 x^13
Printing a random 60 degree polynomial with 8 terms
57.72 x^58 +  42.02 x^52 +  50.18 x^38 +  29.03 x^36 +  6.26 x^28  +  69.76 x^27 +  42.65 x^12
+  45.06
Multiplying monomial in place above with coeff = 68.327217 and power = 43
3943.55 x^101  +  2871.00 x^95   +  3428.90 x^81   +  1983.58 x^79   +  427.52 x^71    +
4766.61 x^70   +  2914.04 x^55   +  3078.70 x^43
Printing a random 70 degree polynomial with 0 terms
0
Multiplying monomial in place above with coeff = 49.703365 and power = 21
0
Printing a random 70 degree polynomial with 2 terms
34.36 x^65 +  5.75 x^22
Multiplying monomial in place above with coeff = 26.137615 and power = 45
898.11 x^110   +  150.27 x^67
Printing a random 70 degree polynomial with 4 terms
```

```
26.14 x^65 +  90.92 x^60 +  18.96 x^47 +  14.48 x^22
Multiplying monomial in place above with coeff = 69.761467 and power = 16
1823.40 x^81   +  6342.96 x^76   +  1322.91 x^63   +  1010.15 x^38
Printing a random 70 degree polynomial with 6 terms
90.92 x^65 +  69.76 x^60 +  45.06 x^54 +  42.65 x^47 +  6.26 x^42  +  18.96 x^22
Multiplying monomial in place above with coeff = 57.715595 and power = 43
5247.71 x^108  +  4026.32 x^103  +  2600.56 x^97   +  2461.47 x^90   +  361.12 x^85    +
1094.48 x^65
Printing a random 70 degree polynomial with 8 terms
69.76 x^65 +  42.02 x^64 +  6.26 x^60  +  29.03 x^54 +  45.06 x^47 +  57.72 x^42 +  50.18 x^33
+  42.65 x^22
Multiplying monomial in place above with coeff = 68.327217 and power = 53
4766.61 x^118  +  2871.00 x^117  +  427.52 x^113   +  1983.58 x^107  +  3078.70 x^100  +
3943.55 x^95   +  3428.90 x^86   +  2914.04 x^75
Printing a random 80 degree polynomial with 0 terms
0
Multiplying monomial in place above with coeff = 49.703365 and power = 51
0
Printing a random 80 degree polynomial with 2 terms
34.36 x^53 +  5.75 x^52
Multiplying monomial in place above with coeff = 26.137615 and power = 15
898.11 x^68    +  150.27 x^67
Printing a random 80 degree polynomial with 4 terms
26.14 x^53 +  14.48 x^52 +  90.92 x^20 +  18.96 x^11
Multiplying monomial in place above with coeff = 69.761467 and power = 26
1823.40 x^79   +  1010.15 x^78   +  6342.96 x^46   +  1322.91 x^37
Printing a random 80 degree polynomial with 6 terms
90.92 x^53 +  18.96 x^52 +  45.06 x^28 +  69.76 x^20 +  42.65 x^11 +  6.26 x^4
Multiplying monomial in place above with coeff = 57.715595 and power = 53
5247.71 x^106  +  1094.48 x^105  +  2600.56 x^81   +  4026.32 x^73   +  2461.47 x^64   +
361.12 x^57
Printing a random 80 degree polynomial with 8 terms
42.02 x^66 +  69.76 x^53 +  42.65 x^52 +  50.18 x^38 +  29.03 x^28 +  6.26 x^20  +  45.06 x^11
+  57.72 x^4
Multiplying monomial in place above with coeff = 68.327217 and power = 3
2871.00 x^69   +  4766.61 x^56   +  2914.04 x^55   +  3428.90 x^41   +  1983.58 x^31   +
427.52 x^23    +  3078.70 x^14   +  3943.55 x^7
Printing a random 90 degree polynomial with 0 terms
0
Multiplying monomial in place above with coeff = 49.703365 and power = 11
0
Printing a random 90 degree polynomial with 2 terms
34.36 x^55 +  5.75 x^12
Multiplying monomial in place above with coeff = 26.137615 and power = 55
898.11 x^110   +  150.27 x^67
Printing a random 90 degree polynomial with 4 terms
26.14 x^55 +  18.96 x^35 +  90.92 x^24 +  14.48 x^12
Multiplying monomial in place above with coeff = 69.761467 and power = 86
1823.40 x^141  +  1322.91 x^121  +  6342.96 x^110  +  1010.15 x^98
Printing a random 90 degree polynomial with 6 terms
90.92 x^55 +  42.65 x^35 +  6.26 x^25  +  69.76 x^24 +  18.96 x^12 +  45.06 x^10
Multiplying monomial in place above with coeff = 57.715595 and power = 43
5247.71 x^98   +  2461.47 x^78   +  361.12 x^68    +  4026.32 x^67   +  1094.48 x^55   +
2600.56 x^53
Printing a random 90 degree polynomial with 8 terms
42.02 x^76 +  50.18 x^73 +  69.76 x^55 +  45.06 x^35 +  57.72 x^25 +  6.26 x^24  +  42.65 x^12
+  29.03 x^10
Multiplying monomial in place above with coeff = 68.327217 and power = 43
2871.00 x^119  +  3428.90 x^116  +  4766.61 x^98   +  3078.70 x^78   +  3943.55 x^68   +
427.52 x^67    +  2914.04 x^55   +  1983.58 x^53
```

5. Testing addition of polynomial

```
~~~~~~~~~~~~~~~~~~~~~
Printing a random 10 degree polynomial with 0 terms
0
```

```
Printing another random 10 degree polynomial with 0 terms
0
~~~~~PRINTING THEIR SUM~~~~~
0
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
Printing a random 10 degree polynomial with 2 terms
34.36 x^6  +  5.75 x^2
Printing another random 10 degree polynomial with 2 terms
27.80 x^10 +  43.41 x^6
~~~~~PRINTING THEIR SUM~~~~~
27.80 x^10 +  77.77 x^6  +  5.75 x^2
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
Printing a random 10 degree polynomial with 4 terms
26.14 x^6  +  18.96 x^3  +  14.48 x^2  +  90.92
Printing another random 10 degree polynomial with 4 terms
13.99 x^10 +  35.66 x^6  +  63.12 x^2  +  75.35 x^1
~~~~~PRINTING THEIR SUM~~~~~
13.99 x^10 +  61.80 x^6  +  18.96 x^3  +  77.60 x^2  +  75.35 x^1  +  90.92
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
Printing a random 10 degree polynomial with 6 terms
90.92 x^6  +  6.26 x^4   +  42.65 x^3  +  18.96 x^2  +  45.06 x^1  +  69.76
Printing another random 10 degree polynomial with 6 terms
63.12 x^10 +  6.59 x^8   +  75.35 x^6  +  89.30 x^4  +  91.82 x^2  +  24.41 x^1
~~~~~PRINTING THEIR SUM~~~~~
63.12 x^10 +  6.59 x^8   +  166.27 x^6 +  95.56 x^4  +  42.65 x^3  +  110.78 x^2 +  69.47 x^1
+  69.76
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
Printing a random 10 degree polynomial with 8 terms
50.18 x^8  +  69.76 x^6  +  42.02 x^5  +  57.72 x^4  +  45.06 x^3  +  42.65 x^2  +  29.03 x^1
+  6.26
Printing another random 10 degree polynomial with 8 terms
91.82 x^10 +  92.74 x^8  +  24.41 x^6  +  63.37 x^4  +  55.92 x^3  +  6.59 x^2   +  89.30 x^1
+  71.53
~~~~~PRINTING THEIR SUM~~~~~
91.82 x^10 +  142.92 x^8 +  94.17 x^6  +  42.02 x^5  +  121.09 x^4 +  100.98 x^3 +  49.24 x^2
+  118.33 x^1 +  77.78
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
Printing a random 20 degree polynomial with 0 terms
0
Printing another random 20 degree polynomial with 0 terms
0
~~~~~PRINTING THEIR SUM~~~~~
0
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
Printing a random 20 degree polynomial with 2 terms
34.36 x^20 +  5.75 x^12
Printing another random 20 degree polynomial with 2 terms
27.80 x^15 +  43.41 x^6
~~~~~PRINTING THEIR SUM~~~~~
34.36 x^20 +  27.80 x^15 +  5.75 x^12  +  43.41 x^6
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
Printing a random 20 degree polynomial with 4 terms
26.14 x^20 +  14.48 x^12 +  18.96 x^11 +  90.92 x^9
Printing another random 20 degree polynomial with 4 terms
13.99 x^15 +  75.35 x^14 +  63.12 x^10 +  35.66 x^6
~~~~~PRINTING THEIR SUM~~~~~
26.14 x^20 +  13.99 x^15 +  75.35 x^14 +  14.48 x^12 +  18.96 x^11 +  63.12 x^10 +  90.92 x^9
+  35.66 x^6
```

```
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
Printing a random 20 degree polynomial with 6 terms
90.92 x^20 +  6.26 x^17  +  18.96 x^12 +  42.65 x^11 +  69.76 x^9  +  45.06
Printing another random 20 degree polynomial with 6 terms
89.30 x^18 +  63.12 x^15 +  24.41 x^14 +  91.82 x^10 +  75.35 x^6  +  6.59 x^1
~~~~~PRINTING THEIR SUM~~~~~
90.92 x^20 +  89.30 x^18 +  6.26 x^17  +  63.12 x^15 +  24.41 x^14 +  18.96 x^12 +  42.65 x^11
+  91.82 x^10 +  69.76 x^9  +  75.35 x^6  +  6.59 x^1   +  45.06
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
Printing a random 20 degree polynomial with 8 terms
69.76 x^20 +  57.72 x^17 +  42.65 x^12 +  45.06 x^11 +  6.26 x^9   +  50.18 x^8  +  42.02 x^5
+  29.03
Printing another random 20 degree polynomial with 8 terms
71.53 x^20 +  63.37 x^18 +  91.82 x^15 +  89.30 x^14 +  6.59 x^10  +  24.41 x^6  +  55.92 x^4
+  92.74 x^1
~~~~~PRINTING THEIR SUM~~~~~
141.29 x^20    +  63.37 x^18 +  57.72 x^17 +  91.82 x^15 +  89.30 x^14 +  42.65 x^12 +  45.06
x^11 +  6.59 x^10  +  6.26 x^9   +  50.18 x^8  +  24.41 x^6  +  42.02 x^5  +  55.92 x^4  +
92.74 x^1  +  29.03
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
Printing a random 30 degree polynomial with 0 terms
0
Printing another random 30 degree polynomial with 0 terms
0
~~~~~PRINTING THEIR SUM~~~~~
0
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
Printing a random 30 degree polynomial with 2 terms
5.75 x^12  +  34.36 x^9
Printing another random 30 degree polynomial with 2 terms
27.80 x^27 +  43.41 x^6
~~~~~PRINTING THEIR SUM~~~~~
27.80 x^27 +  5.75 x^12  +  34.36 x^9  +  43.41 x^6
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
Printing a random 30 degree polynomial with 4 terms
90.92 x^15 +  14.48 x^12 +  26.14 x^9  +  18.96 x^7
Printing another random 30 degree polynomial with 4 terms
75.35 x^30 +  13.99 x^27 +  63.12 x^15 +  35.66 x^6
~~~~~PRINTING THEIR SUM~~~~~
75.35 x^30 +  13.99 x^27 +  154.04 x^15    +  14.48 x^12 +  26.14 x^9  +  18.96 x^7  +  35.66
x^6
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
Printing a random 30 degree polynomial with 6 terms
6.26 x^19  +  69.76 x^15 +  18.96 x^12 +  90.92 x^9  +  45.06 x^8  +  42.65 x^7
Printing another random 30 degree polynomial with 6 terms
24.41 x^30 +  63.12 x^27 +  89.30 x^18 +  6.59 x^16  +  91.82 x^15 +  75.35 x^6
~~~~~PRINTING THEIR SUM~~~~~
24.41 x^30 +  63.12 x^27 +  6.26 x^19  +  89.30 x^18 +  6.59 x^16  +  161.58 x^15    +  18.96
x^12 +  90.92 x^9  +  45.06 x^8  +  42.65 x^7  +  75.35 x^6
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
Printing a random 30 degree polynomial with 8 terms
57.72 x^19 +  42.02 x^16 +  6.26 x^15  +  50.18 x^13 +  42.65 x^12 +  69.76 x^9  +  29.03 x^8
+  45.06 x^7
Printing another random 30 degree polynomial with 8 terms
89.30 x^30 +  91.82 x^27 +  71.53 x^22 +  55.92 x^21 +  63.37 x^18 +  92.74 x^16 +  6.59 x^15
+  24.41 x^6
~~~~~PRINTING THEIR SUM~~~~~
89.30 x^30 +  91.82 x^27 +  71.53 x^22 +  55.92 x^21 +  57.72 x^19 +  63.37 x^18 +  134.76 x^16
```

```
+  12.85 x^15 +  50.18 x^13 +  42.65 x^12 +  69.76 x^9  +  29.03 x^8  +  45.06 x^7  +  24.41
x^6
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
Printing a random 40 degree polynomial with 0 terms
0
Printing another random 40 degree polynomial with 0 terms
0
~~~~~PRINTING THEIR SUM~~~~~
0
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
Printing a random 40 degree polynomial with 2 terms
34.36 x^17 +  5.75 x^12
Printing another random 40 degree polynomial with 2 terms
27.80 x^38 +  43.41 x^6
~~~~~PRINTING THEIR SUM~~~~~
27.80 x^38 +  34.36 x^17 +  5.75 x^12  +  43.41 x^6
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
Printing a random 40 degree polynomial with 4 terms
18.96 x^21 +  26.14 x^17 +  14.48 x^12 +  90.92 x^6
Printing another random 40 degree polynomial with 4 terms
13.99 x^38 +  75.35 x^24 +  35.66 x^6  +  63.12 x^5
~~~~~PRINTING THEIR SUM~~~~~
13.99 x^38 +  75.35 x^24 +  18.96 x^21 +  26.14 x^17 +  14.48 x^12 +  126.58 x^6 +  63.12 x^5
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
Printing a random 40 degree polynomial with 6 terms
45.06 x^24 +  42.65 x^21 +  90.92 x^17 +  18.96 x^12 +  69.76 x^6  +  6.26 x^4
Printing another random 40 degree polynomial with 6 terms
63.12 x^38 +  6.59 x^28  +  24.41 x^24 +  75.35 x^6  +  91.82 x^5  +  89.30
~~~~~PRINTING THEIR SUM~~~~~
63.12 x^38 +  6.59 x^28  +  69.47 x^24 +  42.65 x^21 +  90.92 x^17 +  18.96 x^12 +  145.11 x^6
+  91.82 x^5  +  6.26 x^4   +  89.30
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
Printing a random 40 degree polynomial with 8 terms
29.03 x^24 +  50.18 x^23 +  45.06 x^21 +  42.02 x^20 +  69.76 x^17 +  42.65 x^12 +  6.26 x^6
+  57.72 x^4
Printing another random 40 degree polynomial with 8 terms
91.82 x^38 +  55.92 x^31 +  71.53 x^30 +  92.74 x^28 +  89.30 x^24 +  24.41 x^6  +  6.59 x^5
+  63.37
~~~~~PRINTING THEIR SUM~~~~~
91.82 x^38 +  55.92 x^31 +  71.53 x^30 +  92.74 x^28 +  118.33 x^24    +  50.18 x^23 +  45.06
x^21 +  42.02 x^20 +  69.76 x^17 +  42.65 x^12 +  30.67 x^6  +  6.59 x^5   +  57.72 x^4  +
63.37
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
Printing a random 50 degree polynomial with 0 terms
0
Printing another random 50 degree polynomial with 0 terms
0
~~~~~PRINTING THEIR SUM~~~~~
0
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
Printing a random 50 degree polynomial with 2 terms
5.75 x^42  +  34.36 x^35
Printing another random 50 degree polynomial with 2 terms
43.41 x^36 +  27.80 x^18
~~~~~PRINTING THEIR SUM~~~~~
5.75 x^42  +  43.41 x^36 +  34.36 x^35 +  27.80 x^18
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
```

```
Printing a random 50 degree polynomial with 4 terms
14.48 x^42 +  26.14 x^35 +  90.92 x^17 +  18.96 x^11
Printing another random 50 degree polynomial with 4 terms
75.35 x^38 +  35.66 x^36 +  63.12 x^27 +  13.99 x^18
~~~~~PRINTING THEIR SUM~~~~~
14.48 x^42 +  75.35 x^38 +  35.66 x^36 +  26.14 x^35 +  63.12 x^27 +  13.99 x^18 +  90.92 x^17
+  18.96 x^11
~~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~~~
Printing a random 50 degree polynomial with 6 terms
45.06 x^48 +  6.26 x^44  +  18.96 x^42 +  90.92 x^35 +  69.76 x^17 +  42.65 x^11
Printing another random 50 degree polynomial with 6 terms
24.41 x^38 +  75.35 x^36 +  89.30 x^28 +  91.82 x^27 +  6.59 x^20  +  63.12 x^18
~~~~~PRINTING THEIR SUM~~~~~
45.06 x^48 +  6.26 x^44  +  18.96 x^42 +  24.41 x^38 +  75.35 x^36 +  90.92 x^35 +  89.30 x^28
+  91.82 x^27 +  6.59 x^20  +  63.12 x^18 +  69.76 x^17 +  42.65 x^11
~~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~~~
Printing a random 50 degree polynomial with 8 terms
29.03 x^48 +  57.72 x^44 +  42.65 x^42 +  50.18 x^38 +  69.76 x^35 +  6.26 x^17  +  45.06 x^11
+  42.02 x^5
Printing another random 50 degree polynomial with 8 terms
71.53 x^50 +  89.30 x^38 +  24.41 x^36 +  63.37 x^28 +  6.59 x^27  +  92.74 x^20 +  91.82 x^18
+  55.92 x^1
~~~~~PRINTING THEIR SUM~~~~~
71.53 x^50 +  29.03 x^48 +  57.72 x^44 +  42.65 x^42 +  139.49 x^38    +  24.41 x^36 +  69.76
x^35 +  63.37 x^28 +  6.59 x^27  +  92.74 x^20 +  91.82 x^18 +  6.26 x^17  +  45.06 x^11 +
42.02 x^5  +  55.92 x^1
~~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~~~
Printing a random 60 degree polynomial with 0 terms
0
Printing another random 60 degree polynomial with 0 terms
0
~~~~~PRINTING THEIR SUM~~~~~
0
~~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~~~
Printing a random 60 degree polynomial with 2 terms
34.36 x^27 +  5.75 x^12
Printing another random 60 degree polynomial with 2 terms
27.80 x^59 +  43.41 x^6
~~~~~PRINTING THEIR SUM~~~~~
27.80 x^59 +  34.36 x^27 +  5.75 x^12  +  43.41 x^6
~~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~~~
Printing a random 60 degree polynomial with 4 terms
90.92 x^28 +  26.14 x^27 +  14.48 x^12 +  18.96
Printing another random 60 degree polynomial with 4 terms
13.99 x^59 +  75.35 x^46 +  35.66 x^6  +  63.12 x^1
~~~~~PRINTING THEIR SUM~~~~~
13.99 x^59 +  75.35 x^46 +  90.92 x^28 +  26.14 x^27 +  14.48 x^12 +  35.66 x^6  +  63.12 x^1
+  18.96
~~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~~~
Printing a random 60 degree polynomial with 6 terms
6.26 x^58  +  45.06 x^36 +  69.76 x^28 +  90.92 x^27 +  18.96 x^12 +  42.65
Printing another random 60 degree polynomial with 6 terms
63.12 x^59 +  24.41 x^46 +  6.59 x^19  +  89.30 x^18 +  75.35 x^6  +  91.82 x^1
~~~~~PRINTING THEIR SUM~~~~~
63.12 x^59 +  6.26 x^58  +  24.41 x^46 +  45.06 x^36 +  69.76 x^28 +  90.92 x^27 +  6.59 x^19
+  89.30 x^18 +  18.96 x^12 +  75.35 x^6  +  91.82 x^1  +  42.65
~~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~~~
Printing a random 60 degree polynomial with 8 terms
```

```
57.72 x^58 +  42.02 x^52 +  50.18 x^38 +  29.03 x^36 +  6.26 x^28  +  69.76 x^27 +  42.65 x^12
+  45.06
Printing another random 60 degree polynomial with 8 terms
91.82 x^59 +  89.30 x^46 +  71.53 x^22 +  55.92 x^21 +  92.74 x^19 +  63.37 x^18 +  24.41 x^6
+  6.59 x^1
~~~~~PRINTING THEIR SUM~~~~~
91.82 x^59 +  57.72 x^58 +  42.02 x^52 +  89.30 x^46 +  50.18 x^38 +  29.03 x^36 +  6.26 x^28
+  69.76 x^27 +  71.53 x^22 +  55.92 x^21 +  92.74 x^19 +  63.37 x^18 +  42.65 x^12 +  24.41
x^6  +  6.59 x^1   +  45.06
~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~~
Printing a random 70 degree polynomial with 0 terms
0
Printing another random 70 degree polynomial with 0 terms
0
~~~~~PRINTING THEIR SUM~~~~~
0
~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~~
Printing a random 70 degree polynomial with 2 terms
34.36 x^65 +  5.75 x^22
Printing another random 70 degree polynomial with 2 terms
27.80 x^49 +  43.41 x^6
~~~~~PRINTING THEIR SUM~~~~~
34.36 x^65 +  27.80 x^49 +  5.75 x^22  +  43.41 x^6
~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~~
Printing a random 70 degree polynomial with 4 terms
26.14 x^65 +  90.92 x^60 +  18.96 x^47 +  14.48 x^22
Printing another random 70 degree polynomial with 4 terms
75.35 x^54 +  63.12 x^53 +  13.99 x^49 +  35.66 x^6
~~~~~PRINTING THEIR SUM~~~~~
26.14 x^65 +  90.92 x^60 +  75.35 x^54 +  63.12 x^53 +  13.99 x^49 +  18.96 x^47 +  14.48 x^22
+  35.66 x^6
~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~~
Printing a random 70 degree polynomial with 6 terms
90.92 x^65 +  69.76 x^60 +  45.06 x^54 +  42.65 x^47 +  6.26 x^42  +  18.96 x^22
Printing another random 70 degree polynomial with 6 terms
24.41 x^54 +  91.82 x^53 +  89.30 x^50 +  63.12 x^49 +  6.59 x^23  +  75.35 x^6
~~~~~PRINTING THEIR SUM~~~~~
90.92 x^65 +  69.76 x^60 +  69.47 x^54 +  91.82 x^53 +  89.30 x^50 +  63.12 x^49 +  42.65 x^47
+  6.26 x^42  +  6.59 x^23  +  18.96 x^22 +  75.35 x^6
~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~~
Printing a random 70 degree polynomial with 8 terms
69.76 x^65 +  42.02 x^64 +  6.26 x^60  +  29.03 x^54 +  45.06 x^47 +  57.72 x^42 +  50.18 x^33
+  42.65 x^22
Printing another random 70 degree polynomial with 8 terms
71.53 x^70 +  89.30 x^54 +  6.59 x^53  +  63.37 x^50 +  91.82 x^49 +  55.92 x^41 +  92.74 x^23
+  24.41 x^6
~~~~~PRINTING THEIR SUM~~~~~
71.53 x^70 +  69.76 x^65 +  42.02 x^64 +  6.26 x^60  +  118.33 x^54    +  6.59 x^53  +  63.37
x^50 +  91.82 x^49 +  45.06 x^47 +  57.72 x^42 +  55.92 x^41 +  50.18 x^33 +  92.74 x^23 +
42.65 x^22 +  24.41 x^6
~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~~
Printing a random 80 degree polynomial with 0 terms
0
Printing another random 80 degree polynomial with 0 terms
0
~~~~~PRINTING THEIR SUM~~~~~
0
~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~~
```

```
Printing a random 80 degree polynomial with 2 terms
34.36 x^53 +  5.75 x^52
Printing another random 80 degree polynomial with 2 terms
43.41 x^46 +  27.80 x^39
~~~~~PRINTING THEIR SUM~~~~~
34.36 x^53 +  5.75 x^52  +  43.41 x^46 +  27.80 x^39
~~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~~~
Printing a random 80 degree polynomial with 4 terms
26.14 x^53 +  14.48 x^52 +  90.92 x^20 +  18.96 x^11
Printing another random 80 degree polynomial with 4 terms
75.35 x^80 +  35.66 x^46 +  13.99 x^39 +  63.12 x^7
~~~~~PRINTING THEIR SUM~~~~~
75.35 x^80 +  26.14 x^53 +  14.48 x^52 +  35.66 x^46 +  13.99 x^39 +  90.92 x^20 +  18.96 x^11
+  63.12 x^7
~~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~~~
Printing a random 80 degree polynomial with 6 terms
90.92 x^53 +  18.96 x^52 +  45.06 x^28 +  69.76 x^20 +  42.65 x^11 +  6.26 x^4
Printing another random 80 degree polynomial with 6 terms
24.41 x^80 +  75.35 x^46 +  63.12 x^39 +  89.30 x^38 +  6.59 x^36  +  91.82 x^7
~~~~~PRINTING THEIR SUM~~~~~
24.41 x^80 +  90.92 x^53 +  18.96 x^52 +  75.35 x^46 +  63.12 x^39 +  89.30 x^38 +  6.59 x^36
+  45.06 x^28 +  69.76 x^20 +  42.65 x^11 +  91.82 x^7  +  6.26 x^4
~~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~~~
Printing a random 80 degree polynomial with 8 terms
42.02 x^66 +  69.76 x^53 +  42.65 x^52 +  50.18 x^38 +  29.03 x^28 +  6.26 x^20  +  45.06 x^11
+  57.72 x^4
Printing another random 80 degree polynomial with 8 terms
89.30 x^80 +  71.53 x^78 +  24.41 x^46 +  91.82 x^39 +  63.37 x^38 +  92.74 x^36 +  55.92 x^21
+  6.59 x^7
~~~~~PRINTING THEIR SUM~~~~~
89.30 x^80 +  71.53 x^78 +  42.02 x^66 +  69.76 x^53 +  42.65 x^52 +  24.41 x^46 +  91.82 x^39
+  113.56 x^38    +  92.74 x^36 +  29.03 x^28 +  55.92 x^21 +  6.26 x^20  +  45.06 x^11 +  6.59
x^7   +  57.72 x^4
~~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~~~
Printing a random 90 degree polynomial with 0 terms
0
Printing another random 90 degree polynomial with 0 terms
0
~~~~~PRINTING THEIR SUM~~~~~
0
~~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~~~
Printing a random 90 degree polynomial with 2 terms
34.36 x^55 +  5.75 x^12
Printing another random 90 degree polynomial with 2 terms
43.41 x^66 +  27.80 x^29
~~~~~PRINTING THEIR SUM~~~~~
43.41 x^66 +  34.36 x^55 +  27.80 x^29 +  5.75 x^12
~~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~~~
Printing a random 90 degree polynomial with 4 terms
26.14 x^55 +  18.96 x^35 +  90.92 x^24 +  14.48 x^12
Printing another random 90 degree polynomial with 4 terms
35.66 x^66 +  75.35 x^30 +  13.99 x^29 +  63.12 x^14
~~~~~PRINTING THEIR SUM~~~~~
35.66 x^66 +  26.14 x^55 +  18.96 x^35 +  75.35 x^30 +  13.99 x^29 +  90.92 x^24 +  63.12 x^14
+  14.48 x^12
~~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~~~
Printing a random 90 degree polynomial with 6 terms
90.92 x^55 +  42.65 x^35 +  6.26 x^25  +  69.76 x^24 +  18.96 x^12 +  45.06 x^10
```

```
Printing another random 90 degree polynomial with 6 terms
75.35 x^66 +  89.30 x^56 +  6.59 x^55  +  24.41 x^30 +  63.12 x^29 +  91.82 x^14
~~~~~PRINTING THEIR SUM~~~~~
75.35 x^66 +  89.30 x^56 +  97.51 x^55 +  42.65 x^35 +  24.41 x^30 +  63.12 x^29 +  6.26 x^25
+  69.76 x^24 +  91.82 x^14 +  18.96 x^12 +  45.06 x^10
~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~~
Printing a random 90 degree polynomial with 8 terms
42.02 x^76 +  50.18 x^73 +  69.76 x^55 +  45.06 x^35 +  57.72 x^25 +  6.26 x^24  +  42.65 x^12
+  29.03 x^10
Printing another random 90 degree polynomial with 8 terms
55.92 x^76 +  24.41 x^66 +  63.37 x^56 +  92.74 x^55 +  71.53 x^34 +  89.30 x^30 +  91.82 x^29
+  6.59 x^14
~~~~~PRINTING THEIR SUM~~~~~
97.94 x^76 +  50.18 x^73 +  24.41 x^66 +  63.37 x^56 +  162.50 x^55    +  45.06 x^35 +  71.53
x^34 +  89.30 x^30 +  91.82 x^29 +  57.72 x^25 +  6.26 x^24  +  6.59 x^14  +  42.65 x^12 +
29.03 x^10
~~~~~~~~~~~~~~~~~~~~~
```

6. Testing subtraction of polynomials

```
~~~~~~~~~~~~~~~~~~~~~
Printing a random 10 degree polynomial A with 0 terms
0
Printing another random 10 degree polynomial B with 0 terms
0
~~~~~PRINTING A-B~~~~~
0
~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~~
Printing a random 10 degree polynomial A with 2 terms
34.36 x^6  +  5.75 x^2
Printing another random 10 degree polynomial B with 2 terms
27.80 x^10 +  43.41 x^6
~~~~~PRINTING A-B~~~~~
-27.80 x^10    +  -9.05 x^6  +  5.75 x^2
~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~~
Printing a random 10 degree polynomial A with 4 terms
26.14 x^6  +  18.96 x^3  +  14.48 x^2  +  90.92
Printing another random 10 degree polynomial B with 4 terms
13.99 x^10 +  35.66 x^6  +  63.12 x^2  +  75.35 x^1
~~~~~PRINTING A-B~~~~~
-13.99 x^10    +  -9.52 x^6  +  18.96 x^3  +  -48.64 x^2 +  -75.35 x^1 +  90.92
~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~~
Printing a random 10 degree polynomial A with 6 terms
90.92 x^6  +  6.26 x^4   +  42.65 x^3  +  18.96 x^2  +  45.06 x^1  +  69.76
Printing another random 10 degree polynomial B with 6 terms
63.12 x^10 +  6.59 x^8   +  75.35 x^6  +  89.30 x^4  +  91.82 x^2  +  24.41 x^1
~~~~~PRINTING A-B~~~~~
-63.12 x^10    +  -6.59 x^8  +  15.57 x^6  +  -83.05 x^4 +  42.65 x^3  +  -72.85 x^2 +  20.65
x^1  +  69.76
~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~~
Printing a random 10 degree polynomial A with 8 terms
50.18 x^8  +  69.76 x^6  +  42.02 x^5  +  57.72 x^4  +  45.06 x^3  +  42.65 x^2  +  29.03 x^1
+  6.26
Printing another random 10 degree polynomial B with 8 terms
91.82 x^10 +  92.74 x^8  +  24.41 x^6  +  63.37 x^4  +  55.92 x^3  +  6.59 x^2   +  89.30 x^1
+  71.53
~~~~~PRINTING A-B~~~~~
-91.82 x^10    +  -42.56 x^8 +  45.35 x^6  +  42.02 x^5  +  -5.66 x^4  +  -10.86 x^3 +  36.06
x^2  +  -60.27 x^1 +  -65.27
~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~~
```

```
Printing a random 20 degree polynomial A with 0 terms
0
Printing another random 20 degree polynomial B with 0 terms
0
~~~~~PRINTING A-B~~~~~
0
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
Printing a random 20 degree polynomial A with 2 terms
34.36 x^20 +  5.75 x^12
Printing another random 20 degree polynomial B with 2 terms
27.80 x^15 +  43.41 x^6
~~~~~PRINTING A-B~~~~~
34.36 x^20 +  -27.80 x^15    +  5.75 x^12  +  -43.41 x^6
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
Printing a random 20 degree polynomial A with 4 terms
26.14 x^20 +  14.48 x^12 +  18.96 x^11 +  90.92 x^9
Printing another random 20 degree polynomial B with 4 terms
13.99 x^15 +  75.35 x^14 +  63.12 x^10 +  35.66 x^6
~~~~~PRINTING A-B~~~~~
26.14 x^20 +  -13.99 x^15    +  -75.35 x^14    +  14.48 x^12 +  18.96 x^11 +  -63.12 x^10    +
90.92 x^9  +  -35.66 x^6
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
Printing a random 20 degree polynomial A with 6 terms
90.92 x^20 +  6.26 x^17  +  18.96 x^12 +  42.65 x^11 +  69.76 x^9  +  45.06
Printing another random 20 degree polynomial B with 6 terms
89.30 x^18 +  63.12 x^15 +  24.41 x^14 +  91.82 x^10 +  75.35 x^6  +  6.59 x^1
~~~~~PRINTING A-B~~~~~
90.92 x^20 +  -89.30 x^18    +  6.26 x^17  +  -63.12 x^15    +  -24.41 x^14    +  18.96 x^12 +
42.65 x^11 +  -91.82 x^10    +  69.76 x^9  +  -75.35 x^6 +  -6.59 x^1  +  45.06
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
Printing a random 20 degree polynomial A with 8 terms
69.76 x^20 +  57.72 x^17 +  42.65 x^12 +  45.06 x^11 +  6.26 x^9   +  50.18 x^8  +  42.02 x^5
+  29.03
Printing another random 20 degree polynomial B with 8 terms
71.53 x^20 +  63.37 x^18 +  91.82 x^15 +  89.30 x^14 +  6.59 x^10  +  24.41 x^6  +  55.92 x^4
+  92.74 x^1
~~~~~PRINTING A-B~~~~~
-1.76 x^20 +  -63.37 x^18    +  57.72 x^17 +  -91.82 x^15    +  -89.30 x^14    +  42.65 x^12 +
45.06 x^11 +  -6.59 x^10 +  6.26 x^9   +  50.18 x^8  +  -24.41 x^6 +  42.02 x^5  +  -55.92 x^4
+  -92.74 x^1 +  29.03
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
Printing a random 30 degree polynomial A with 0 terms
0
Printing another random 30 degree polynomial B with 0 terms
0
~~~~~PRINTING A-B~~~~~
0
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
Printing a random 30 degree polynomial A with 2 terms
5.75 x^12  +  34.36 x^9
Printing another random 30 degree polynomial B with 2 terms
27.80 x^27 +  43.41 x^6
~~~~~PRINTING A-B~~~~~
-27.80 x^27    +  5.75 x^12  +  34.36 x^9  +  -43.41 x^6
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
Printing a random 30 degree polynomial A with 4 terms
90.92 x^15 +  14.48 x^12 +  26.14 x^9  +  18.96 x^7
Printing another random 30 degree polynomial B with 4 terms
```

```
75.35 x^30 +  13.99 x^27 +  63.12 x^15 +  35.66 x^6
~~~~~PRINTING A-B~~~~~
-75.35 x^30    +  -13.99 x^27    +  27.80 x^15 +  14.48 x^12 +  26.14 x^9  +  18.96 x^7  +  -
35.66 x^6
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
Printing a random 30 degree polynomial A with 6 terms
6.26 x^19  +  69.76 x^15 +  18.96 x^12 +  90.92 x^9  +  45.06 x^8  +  42.65 x^7
Printing another random 30 degree polynomial B with 6 terms
24.41 x^30 +  63.12 x^27 +  89.30 x^18 +  6.59 x^16  +  91.82 x^15 +  75.35 x^6
~~~~~PRINTING A-B~~~~~
-24.41 x^30    +  -63.12 x^27    +  6.26 x^19  +  -89.30 x^18    +  -6.59 x^16 +  -22.06 x^15
+  18.96 x^12 +  90.92 x^9  +  45.06 x^8  +  42.65 x^7  +  -75.35 x^6
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
Printing a random 30 degree polynomial A with 8 terms
57.72 x^19 +  42.02 x^16 +  6.26 x^15  +  50.18 x^13 +  42.65 x^12 +  69.76 x^9  +  29.03 x^8
+  45.06 x^7
Printing another random 30 degree polynomial B with 8 terms
89.30 x^30 +  91.82 x^27 +  71.53 x^22 +  55.92 x^21 +  63.37 x^18 +  92.74 x^16 +  6.59 x^15
+  24.41 x^6
~~~~~PRINTING A-B~~~~~
-89.30 x^30    +  -91.82 x^27    +  -71.53 x^22    +  -55.92 x^21    +  57.72 x^19 +  -63.37
x^18    +  -50.72 x^16    +  -0.33 x^15 +  50.18 x^13 +  42.65 x^12 +  69.76 x^9  +  29.03 x^8
+  45.06 x^7  +  -24.41 x^6
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
Printing a random 40 degree polynomial A with 0 terms
0
Printing another random 40 degree polynomial B with 0 terms
0
~~~~~PRINTING A-B~~~~~
0
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
Printing a random 40 degree polynomial A with 2 terms
34.36 x^17 +  5.75 x^12
Printing another random 40 degree polynomial B with 2 terms
27.80 x^38 +  43.41 x^6
~~~~~PRINTING A-B~~~~~
-27.80 x^38    +  34.36 x^17 +  5.75 x^12  +  -43.41 x^6
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
Printing a random 40 degree polynomial A with 4 terms
18.96 x^21 +  26.14 x^17 +  14.48 x^12 +  90.92 x^6
Printing another random 40 degree polynomial B with 4 terms
13.99 x^38 +  75.35 x^24 +  35.66 x^6  +  63.12 x^5
~~~~~PRINTING A-B~~~~~
-13.99 x^38    +  -75.35 x^24    +  18.96 x^21 +  26.14 x^17 +  14.48 x^12 +  55.26 x^6  +  -
63.12 x^5
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
Printing a random 40 degree polynomial A with 6 terms
45.06 x^24 +  42.65 x^21 +  90.92 x^17 +  18.96 x^12 +  69.76 x^6  +  6.26 x^4
Printing another random 40 degree polynomial B with 6 terms
63.12 x^38 +  6.59 x^28  +  24.41 x^24 +  75.35 x^6  +  91.82 x^5  +  89.30
~~~~~PRINTING A-B~~~~~
-63.12 x^38    +  -6.59 x^28 +  20.65 x^24 +  42.65 x^21 +  90.92 x^17 +  18.96 x^12 +  -5.59
x^6  +  -91.82 x^5 +  6.26 x^4   +  -89.30
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
Printing a random 40 degree polynomial A with 8 terms
29.03 x^24 +  50.18 x^23 +  45.06 x^21 +  42.02 x^20 +  69.76 x^17 +  42.65 x^12 +  6.26 x^6
+  57.72 x^4
Printing another random 40 degree polynomial B with 8 terms
```

```
91.82 x^38 +  55.92 x^31 +  71.53 x^30 +  92.74 x^28 +  89.30 x^24 +  24.41 x^6  +  6.59 x^5
+  63.37
~~~~~PRINTING A-B~~~~~
-91.82 x^38    +  -55.92 x^31    +  -71.53 x^30    +  -92.74 x^28    +  -60.27 x^24    +  50.18
x^23 +  45.06 x^21 +  42.02 x^20 +  69.76 x^17 +  42.65 x^12 +  -18.15 x^6 +  -6.59 x^5  +
57.72 x^4  +  -63.37
~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~~
Printing a random 50 degree polynomial A with 0 terms
0
Printing another random 50 degree polynomial B with 0 terms
0
~~~~~PRINTING A-B~~~~~
0
~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~~
Printing a random 50 degree polynomial A with 2 terms
5.75 x^42  +  34.36 x^35
Printing another random 50 degree polynomial B with 2 terms
43.41 x^36 +  27.80 x^18
~~~~~PRINTING A-B~~~~~
5.75 x^42  +  -43.41 x^36    +  34.36 x^35 +  -27.80 x^18
~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~~
Printing a random 50 degree polynomial A with 4 terms
14.48 x^42 +  26.14 x^35 +  90.92 x^17 +  18.96 x^11
Printing another random 50 degree polynomial B with 4 terms
75.35 x^38 +  35.66 x^36 +  63.12 x^27 +  13.99 x^18
~~~~~PRINTING A-B~~~~~
14.48 x^42 +  -75.35 x^38    +  -35.66 x^36    +  26.14 x^35 +  -63.12 x^27    +  -13.99 x^18
+  90.92 x^17 +  18.96 x^11
~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~~
Printing a random 50 degree polynomial A with 6 terms
45.06 x^48 +  6.26 x^44  +  18.96 x^42 +  90.92 x^35 +  69.76 x^17 +  42.65 x^11
Printing another random 50 degree polynomial B with 6 terms
24.41 x^38 +  75.35 x^36 +  89.30 x^28 +  91.82 x^27 +  6.59 x^20  +  63.12 x^18
~~~~~PRINTING A-B~~~~~
45.06 x^48 +  6.26 x^44  +  18.96 x^42 +  -24.41 x^38    +  -75.35 x^36    +  90.92 x^35 +  -
89.30 x^28    +  -91.82 x^27    +  -6.59 x^20 +  -63.12 x^18    +  69.76 x^17 +  42.65 x^11
~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~~
Printing a random 50 degree polynomial A with 8 terms
29.03 x^48 +  57.72 x^44 +  42.65 x^42 +  50.18 x^38 +  69.76 x^35 +  6.26 x^17  +  45.06 x^11
+  42.02 x^5
Printing another random 50 degree polynomial B with 8 terms
71.53 x^50 +  89.30 x^38 +  24.41 x^36 +  63.37 x^28 +  6.59 x^27  +  92.74 x^20 +  91.82 x^18
+  55.92 x^1
~~~~~PRINTING A-B~~~~~
-71.53 x^50    +  29.03 x^48 +  57.72 x^44 +  42.65 x^42 +  -39.12 x^38    +  -24.41 x^36    +
69.76 x^35 +  -63.37 x^28    +  -6.59 x^27 +  -92.74 x^20    +  -91.82 x^18    +  6.26 x^17  +
45.06 x^11 +  42.02 x^5  +  -55.92 x^1
~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~~
Printing a random 60 degree polynomial A with 0 terms
0
Printing another random 60 degree polynomial B with 0 terms
0
~~~~~PRINTING A-B~~~~~
0
~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~~
Printing a random 60 degree polynomial A with 2 terms
34.36 x^27 +  5.75 x^12
Printing another random 60 degree polynomial B with 2 terms
```

```
27.80 x^59 +  43.41 x^6
~~~~~PRINTING A-B~~~~~
-27.80 x^59    +  34.36 x^27 +  5.75 x^12  +  -43.41 x^6
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
Printing a random 60 degree polynomial A with 4 terms
90.92 x^28 +  26.14 x^27 +  14.48 x^12 +  18.96
Printing another random 60 degree polynomial B with 4 terms
13.99 x^59 +  75.35 x^46 +  35.66 x^6  +  63.12 x^1
~~~~~PRINTING A-B~~~~~
-13.99 x^59    +  -75.35 x^46    +  90.92 x^28 +  26.14 x^27 +  14.48 x^12 +  -35.66 x^6 +  -
63.12 x^1 +  18.96
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
Printing a random 60 degree polynomial A with 6 terms
6.26 x^58  +  45.06 x^36 +  69.76 x^28 +  90.92 x^27 +  18.96 x^12 +  42.65
Printing another random 60 degree polynomial B with 6 terms
63.12 x^59 +  24.41 x^46 +  6.59 x^19  +  89.30 x^18 +  75.35 x^6  +  91.82 x^1
~~~~~PRINTING A-B~~~~~
-63.12 x^59    +  6.26 x^58  +  -24.41 x^46    +  45.06 x^36 +  69.76 x^28 +  90.92 x^27 +  -
6.59 x^19 +  -89.30 x^18    +  18.96 x^12 +  -75.35 x^6 +  -91.82 x^1 +  42.65
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
Printing a random 60 degree polynomial A with 8 terms
57.72 x^58 +  42.02 x^52 +  50.18 x^38 +  29.03 x^36 +  6.26 x^28  +  69.76 x^27 +  42.65 x^12
+  45.06
Printing another random 60 degree polynomial B with 8 terms
91.82 x^59 +  89.30 x^46 +  71.53 x^22 +  55.92 x^21 +  92.74 x^19 +  63.37 x^18 +  24.41 x^6
+  6.59 x^1
~~~~~PRINTING A-B~~~~~
-91.82 x^59    +  57.72 x^58 +  42.02 x^52 +  -89.30 x^46    +  50.18 x^38 +  29.03 x^36 +
6.26 x^28  +  69.76 x^27 +  -71.53 x^22    +  -55.92 x^21    +  -92.74 x^19    +  -63.37 x^18
+  42.65 x^12 +  -24.41 x^6 +  -6.59 x^1  +  45.06
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
Printing a random 70 degree polynomial A with 0 terms
0
Printing another random 70 degree polynomial B with 0 terms
0
~~~~~PRINTING A-B~~~~~
0
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
Printing a random 70 degree polynomial A with 2 terms
34.36 x^65 +  5.75 x^22
Printing another random 70 degree polynomial B with 2 terms
27.80 x^49 +  43.41 x^6
~~~~~PRINTING A-B~~~~~
34.36 x^65 +  -27.80 x^49    +  5.75 x^22  +  -43.41 x^6
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
Printing a random 70 degree polynomial A with 4 terms
26.14 x^65 +  90.92 x^60 +  18.96 x^47 +  14.48 x^22
Printing another random 70 degree polynomial B with 4 terms
75.35 x^54 +  63.12 x^53 +  13.99 x^49 +  35.66 x^6
~~~~~PRINTING A-B~~~~~
26.14 x^65 +  90.92 x^60 +  -75.35 x^54    +  -63.12 x^53    +  -13.99 x^49    +  18.96 x^47 +
14.48 x^22 +  -35.66 x^6
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
Printing a random 70 degree polynomial A with 6 terms
90.92 x^65 +  69.76 x^60 +  45.06 x^54 +  42.65 x^47 +  6.26 x^42  +  18.96 x^22
Printing another random 70 degree polynomial B with 6 terms
24.41 x^54 +  91.82 x^53 +  89.30 x^50 +  63.12 x^49 +  6.59 x^23  +  75.35 x^6
~~~~~PRINTING A-B~~~~~
```

```
90.92 x^65 +  69.76 x^60 +  20.65 x^54 +  -91.82 x^53    +  -89.30 x^50    +  -63.12 x^49    +
42.65 x^47 +  6.26 x^42  +  -6.59 x^23 +  18.96 x^22 +  -75.35 x^6
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
Printing a random 70 degree polynomial A with 8 terms
69.76 x^65 +  42.02 x^64 +  6.26 x^60  +  29.03 x^54 +  45.06 x^47 +  57.72 x^42 +  50.18 x^33
+  42.65 x^22
Printing another random 70 degree polynomial B with 8 terms
71.53 x^70 +  89.30 x^54 +  6.59 x^53  +  63.37 x^50 +  91.82 x^49 +  55.92 x^41 +  92.74 x^23
+  24.41 x^6
~~~~~PRINTING A-B~~~~~
-71.53 x^70    +  69.76 x^65 +  42.02 x^64 +  6.26 x^60  +  -60.27 x^54    +  -6.59 x^53 +  -
63.37 x^50    +  -91.82 x^49    +  45.06 x^47 +  57.72 x^42 +  -55.92 x^41    +  50.18 x^33 +
-92.74 x^23    +  42.65 x^22 +  -24.41 x^6
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
Printing a random 80 degree polynomial A with 0 terms
0
Printing another random 80 degree polynomial B with 0 terms
0
~~~~~PRINTING A-B~~~~~
0
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
Printing a random 80 degree polynomial A with 2 terms
34.36 x^53 +  5.75 x^52
Printing another random 80 degree polynomial B with 2 terms
43.41 x^46 +  27.80 x^39
~~~~~PRINTING A-B~~~~~
34.36 x^53 +  5.75 x^52  +  -43.41 x^46    +  -27.80 x^39
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
Printing a random 80 degree polynomial A with 4 terms
26.14 x^53 +  14.48 x^52 +  90.92 x^20 +  18.96 x^11
Printing another random 80 degree polynomial B with 4 terms
75.35 x^80 +  35.66 x^46 +  13.99 x^39 +  63.12 x^7
~~~~~PRINTING A-B~~~~~
-75.35 x^80    +  26.14 x^53 +  14.48 x^52 +  -35.66 x^46    +  -13.99 x^39    +  90.92 x^20 +
18.96 x^11 +  -63.12 x^7
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
Printing a random 80 degree polynomial A with 6 terms
90.92 x^53 +  18.96 x^52 +  45.06 x^28 +  69.76 x^20 +  42.65 x^11 +  6.26 x^4
Printing another random 80 degree polynomial B with 6 terms
24.41 x^80 +  75.35 x^46 +  63.12 x^39 +  89.30 x^38 +  6.59 x^36  +  91.82 x^7
~~~~~PRINTING A-B~~~~~
-24.41 x^80    +  90.92 x^53 +  18.96 x^52 +  -75.35 x^46    +  -63.12 x^39    +  -89.30 x^38
+  -6.59 x^36 +  45.06 x^28 +  69.76 x^20 +  42.65 x^11 +  -91.82 x^7 +  6.26 x^4
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
Printing a random 80 degree polynomial A with 8 terms
42.02 x^66 +  69.76 x^53 +  42.65 x^52 +  50.18 x^38 +  29.03 x^28 +  6.26 x^20  +  45.06 x^11
+  57.72 x^4
Printing another random 80 degree polynomial B with 8 terms
89.30 x^80 +  71.53 x^78 +  24.41 x^46 +  91.82 x^39 +  63.37 x^38 +  92.74 x^36 +  55.92 x^21
+  6.59 x^7
~~~~~PRINTING A-B~~~~~
-89.30 x^80    +  -71.53 x^78    +  42.02 x^66 +  69.76 x^53 +  42.65 x^52 +  -24.41 x^46    +
-91.82 x^39    +  -13.19 x^38    +  -92.74 x^36    +  29.03 x^28 +  -55.92 x^21    +  6.26 x^20
+  45.06 x^11 +  -6.59 x^7  +  57.72 x^4
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
Printing a random 90 degree polynomial A with 0 terms
0
Printing another random 90 degree polynomial B with 0 terms
```

```
0
~~~~~PRINTING A-B~~~~~
0
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
Printing a random 90 degree polynomial A with 2 terms
34.36 x^55 +  5.75 x^12
Printing another random 90 degree polynomial B with 2 terms
43.41 x^66 +  27.80 x^29
~~~~~PRINTING A-B~~~~~
-43.41 x^66    +  34.36 x^55 +  -27.80 x^29    +  5.75 x^12
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
Printing a random 90 degree polynomial A with 4 terms
26.14 x^55 +  18.96 x^35 +  90.92 x^24 +  14.48 x^12
Printing another random 90 degree polynomial B with 4 terms
35.66 x^66 +  75.35 x^30 +  13.99 x^29 +  63.12 x^14
~~~~~PRINTING A-B~~~~~
-35.66 x^66    +  26.14 x^55 +  18.96 x^35 +  -75.35 x^30    +  -13.99 x^29    +  90.92 x^24 +
-63.12 x^14    +  14.48 x^12
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
Printing a random 90 degree polynomial A with 6 terms
90.92 x^55 +  42.65 x^35 +  6.26 x^25  +  69.76 x^24 +  18.96 x^12 +  45.06 x^10
Printing another random 90 degree polynomial B with 6 terms
75.35 x^66 +  89.30 x^56 +  6.59 x^55  +  24.41 x^30 +  63.12 x^29 +  91.82 x^14
~~~~~PRINTING A-B~~~~~
-75.35 x^66    +  -89.30 x^56    +  84.33 x^55 +  42.65 x^35 +  -24.41 x^30    +  -63.12 x^29
+  6.26 x^25  +  69.76 x^24 +  -91.82 x^14    +  18.96 x^12 +  45.06 x^10
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
Printing a random 90 degree polynomial A with 8 terms
42.02 x^76 +  50.18 x^73 +  69.76 x^55 +  45.06 x^35 +  57.72 x^25 +  6.26 x^24  +  42.65 x^12
+  29.03 x^10
Printing another random 90 degree polynomial B with 8 terms
55.92 x^76 +  24.41 x^66 +  63.37 x^56 +  92.74 x^55 +  71.53 x^34 +  89.30 x^30 +  91.82 x^29
+  6.59 x^14
~~~~~PRINTING A-B~~~~~
-13.90 x^76    +  50.18 x^73 +  -24.41 x^66    +  -63.37 x^56    +  -22.98 x^55    +  45.06
x^35 +  -71.53 x^34    +  -89.30 x^30    +  -91.82 x^29    +  57.72 x^25 +  6.26 x^24  +  -6.59
x^14 +  42.65 x^12 +  29.03 x^10
~~~~~~~~~~~~~~~~~~~~

## 7. Testing division of polynomials
~~~~~~~~~~~~~~~~~~~~
~~~~~PRINTING A/B~~~~~
A  =  47.37 x^5
B  =  43.69
   x  1.08 x^5
   +  0
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
~~~~~PRINTING A/B~~~~~
A  =  14.48 x^8  +  5.75 x^5   +  26.14 x^4
B  =  13.99 x^8  +  35.66 x^2  +  43.41
   x  1.03
   +  5.75 x^5   +  26.14 x^4  +  -36.91 x^2 +  -44.93
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
~~~~~PRINTING A/B~~~~~
A  =  69.76 x^11 +  18.96 x^8  +  42.65 x^7  +  14.48 x^5  +  90.92 x^4
B  =  24.41 x^11 +  63.12 x^8  +  91.82 x^5  +  75.35 x^2  +  35.66
   x  2.86
   +  -161.43 x^8    +  42.65 x^7  +  -247.93 x^5    +  90.92 x^4  +  -215.34 x^2    +  -101.92
~~~~~~~~~~~~~~~~~~~~
```

```
~~~~~~~~~~~~~~~~~~~~
~~~~~PRINTING A/B~~~~~
A  =  6.26 x^11  +  42.65 x^8  +  45.06 x^7  +  18.96 x^5  +  69.76 x^4  +  57.72 x^2  +  29.03
B  =  89.30 x^11 +  91.82 x^8  +  63.37 x^6  +  6.59 x^5   +  92.74 x^4  +  24.41 x^2  +  75.35
   x  0.07
   +  36.22 x^8  +  45.06 x^7  +  -4.44 x^6  +  18.50 x^5  +  63.26 x^4  +  56.01 x^2  +  23.75
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
~~~~~PRINTING A/B~~~~~
A  =  57.72 x^11 +  45.06 x^8  +  29.03 x^7  +  68.33 x^6  +  42.65 x^5  +  6.26 x^4   +  50.18
x^2  +  32.06 x^1  +  42.02
B  =  63.37 x^11 +  6.59 x^8   +  71.53 x^6  +  92.74 x^5  +  55.92 x^4  +  73.50 x^3  +  89.30
x^2  +  86.13 x^1  +  24.41
   x  0.91
   +  39.06 x^8  +  29.03 x^7  +  3.19 x^6   +  -41.81 x^5 +  -44.67 x^4 +  -66.94 x^3 +  -
31.15 x^2 +  -46.38 x^1 +  19.79
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
~~~~~PRINTING A/B~~~~~
A  =  47.37 x^17
B  =  43.69 x^10
   x  1.08 x^7
   +  0
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
~~~~~PRINTING A/B~~~~~
A  =  14.48 x^18 +  5.75 x^17  +  26.14 x^15
B  =  35.66 x^12 +  43.41 x^10 +  13.99 x^6
   x  0.41 x^6   +  0.16 x^5   +  -0.49 x^4  +  0.54 x^3   +  0.60 x^2   +  -0.65 x^1  +  -0.89
   +  26.11 x^11 +  45.63 x^10 +  -7.51 x^9  +  -8.42 x^8  +  9.14 x^7   +  12.48 x^6
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
~~~~~PRINTING A/B~~~~~
A  =  69.76 x^21 +  42.65 x^19 +  18.96 x^18 +  14.48 x^17 +  90.92 x^15
B  =  75.35 x^12 +  35.66 x^10 +  24.41 x^7  +  63.12 x^6  +  91.82 x^5
   x  0.93 x^9   +  0.13 x^7   +  0.25 x^6   +  0.13 x^5   +  -0.42 x^4  +  0.37 x^3   +  -0.97
x^2  +  -0.36 x^1  +  0.05
   +  -8.24 x^11 +  3.56 x^10  +  38.91 x^9  +  36.31 x^8  +  110.87 x^7 +  30.16 x^6  +  -4.63
x^5
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
~~~~~PRINTING A/B~~~~~
A  =  6.26 x^21  +  45.06 x^19 +  42.65 x^18 +  18.96 x^17 +  69.76 x^15 +  57.72 x^14 +  29.03
x^10
B  =  92.74 x^21 +  24.41 x^12 +  75.35 x^10 +  89.30 x^7  +  91.82 x^6  +  6.59 x^5   +  63.37
x^1
   x  0.07
   +  45.06 x^19 +  42.65 x^18 +  18.96 x^17 +  69.76 x^15 +  57.72 x^14 +  -1.65 x^12 +  23.95
x^10 +  -6.02 x^7  +  -6.19 x^6  +  -0.44 x^5  +  -4.28 x^1
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
~~~~~PRINTING A/B~~~~~
A  =  57.72 x^21 +  29.03 x^19 +  45.06 x^18 +  42.65 x^17 +  6.26 x^15  +  50.18 x^14 +  68.33
x^13 +  42.02 x^10 +  32.06 x^5
B  =  55.92 x^21 +  73.50 x^16 +  89.30 x^12 +  24.41 x^10 +  63.37 x^7  +  6.59 x^6   +  92.74
x^5  +  86.13 x^4  +  71.53 x^1
   x  1.03
   +  29.03 x^19 +  45.06 x^18 +  42.65 x^17 +  -75.86 x^16    +  6.26 x^15  +  50.18 x^14 +
68.33 x^13 +  -92.17 x^12    +  16.82 x^10 +  -65.41 x^7 +  -6.80 x^6  +  -63.66 x^5 +  -88.89
x^4 +  -73.83 x^1
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
~~~~~PRINTING A/B~~~~~
A  =  47.37 x^29
B  =  43.69 x^24
```

```
   x  1.08 x^5
   +  0
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
~~~~~PRINTING A/B~~~~~
A  =  5.75 x^29  +  26.14 x^23 +  14.48 x^18
B  =  13.99 x^26 +  43.41 x^24 +  35.66 x^2
   x  0.41 x^3   +  -1.28 x^1
   +  55.35 x^25 +  26.14 x^23 +  14.48 x^18 +  -14.65 x^5 +  45.47 x^3
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
~~~~~PRINTING A/B~~~~~
A  =  14.48 x^29 +  69.76 x^27 +  90.92 x^23 +  18.96 x^18 +  42.65 x^11
B  =  63.12 x^26 +  35.66 x^24 +  91.82 x^17 +  24.41 x^15 +  75.35 x^2
   x  0.23 x^3   +  0.98 x^1
   +  -34.79 x^25    +  90.92 x^23 +  -21.06 x^20    +  -76.21 x^18    +  -23.81 x^16    +
42.65 x^11 +  -17.29 x^5 +  -73.51 x^3
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
~~~~~PRINTING A/B~~~~~
A  =  18.96 x^29 +  6.26 x^27  +  69.76 x^23 +  29.03 x^20 +  42.65 x^18 +  57.72 x^15 +  45.06
x^11
B  =  92.74 x^29 +  63.37 x^28 +  91.82 x^26 +  75.35 x^24 +  6.59 x^17  +  89.30 x^15 +  24.41
x^2
   x  0.20
   +  -12.96 x^28    +  6.26 x^27  +  -18.77 x^26    +  -15.41 x^24    +  69.76 x^23 +  29.03
x^20 +  42.65 x^18 +  -1.35 x^17 +  39.46 x^15 +  45.06 x^11 +  -4.99 x^2
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
~~~~~PRINTING A/B~~~~~
A  =  42.65 x^29 +  57.72 x^27 +  6.26 x^23  +  42.02 x^20 +  45.06 x^18 +  50.18 x^15 +  29.03
x^11 +  68.33 x^8  +  32.06 x^6
B  =  55.92 x^29 +  71.53 x^28 +  6.59 x^26  +  24.41 x^24 +  73.50 x^21 +  92.74 x^17 +  63.37
x^15 +  86.13 x^8  +  89.30 x^2
   x  0.76
   +  -54.55 x^28    +  57.72 x^27 +  -5.03 x^26 +  -18.62 x^24    +  6.26 x^23  +  -56.06 x^21
+  42.02 x^20 +  45.06 x^18 +  -70.73 x^17    +  1.85 x^15  +  29.03 x^11 +  2.64 x^8   +
32.06 x^6  +  -68.11 x^2
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
~~~~~PRINTING A/B~~~~~
A  =  47.37 x^41
B  =  43.69 x^36
   x  1.08 x^5
   +  0
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
~~~~~PRINTING A/B~~~~~
A  =  5.75 x^41  +  14.48 x^38 +  26.14 x^35
B  =  43.41 x^36 +  13.99 x^18 +  35.66 x^12
   x  0.13 x^5   +  0.33 x^2
   +  26.14 x^35 +  -1.85 x^23 +  -4.67 x^20 +  -4.72 x^17 +  -11.90 x^14
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
~~~~~PRINTING A/B~~~~~
A  =  14.48 x^41 +  42.65 x^39 +  18.96 x^38 +  90.92 x^35 +  69.76 x^11
B  =  91.82 x^41 +  35.66 x^36 +  24.41 x^19 +  63.12 x^18 +  75.35 x^12
   x  0.16
   +  42.65 x^39 +  18.96 x^38 +  -5.62 x^36 +  90.92 x^35 +  -3.85 x^19 +  -9.95 x^18 +  -
11.88 x^12    +  69.76 x^11
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
~~~~~PRINTING A/B~~~~~
A  =  18.96 x^41 +  45.06 x^39 +  42.65 x^38 +  69.76 x^35 +  6.26 x^11  +  29.03 x^2  +  57.72
B  =  6.59 x^41  +  75.35 x^36 +  89.30 x^19 +  91.82 x^18 +  24.41 x^12 +  92.74 x^11 +  63.37
```

```
x^1
   x  2.88
   +  45.06 x^39 +  42.65 x^38 +  -216.82 x^36   +  69.76 x^35 +  -256.97 x^19   +  -264.20
x^18   +  -70.24 x^12    +  -260.60 x^11   +  29.03 x^2  +  -182.36 x^1    +  57.72
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
~~~~~PRINTING A/B~~~~~
A  =  42.65 x^41 +  32.06 x^40 +  29.03 x^39 +  45.06 x^38 +  6.26 x^35  +  68.33 x^13 +  57.72
x^11 +  42.02 x^2  +  50.18
B  =  92.74 x^41 +  24.41 x^36 +  63.37 x^19 +  6.59 x^18  +  89.30 x^12 +  55.92 x^11 +  86.13
x^10 +  73.50 x^3  +  71.53 x^1
   x  0.46
   +  32.06 x^40 +  29.03 x^39 +  45.06 x^38 +  -11.23 x^36    +  6.26 x^35  +  -29.14 x^19
+  -3.03 x^18 +  68.33 x^13 +  -41.07 x^12    +  32.00 x^11 +  -39.61 x^10    +  -33.80 x^3 +
42.02 x^2  +  -32.89 x^1 +  50.18
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
~~~~~PRINTING A/B~~~~~
A  =  47.37 x^29
B  =  43.69 x^16
   x  1.08 x^13
   +  0
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
~~~~~PRINTING A/B~~~~~
A  =  26.14 x^39 +  14.48 x^38 +  5.75 x^29
B  =  35.66 x^42 +  43.41 x^16 +  13.99 x^6
   x  0
   +  26.14 x^39 +  14.48 x^38 +  5.75 x^29
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
~~~~~PRINTING A/B~~~~~
A  =  90.92 x^39 +  18.96 x^38 +  14.48 x^29 +  69.76 x^21 +  42.65 x^7
B  =  75.35 x^42 +  91.82 x^37 +  24.41 x^19 +  35.66 x^16 +  63.12 x^6
   x  0
   +  90.92 x^39 +  18.96 x^38 +  14.48 x^29 +  69.76 x^21 +  42.65 x^7
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
~~~~~PRINTING A/B~~~~~
A  =  57.72 x^40 +  69.76 x^39 +  42.65 x^38 +  18.96 x^29 +  29.03 x^22 +  6.26 x^21  +  45.06
x^7
B  =  24.41 x^42 +  63.37 x^38 +  6.59 x^37  +  89.30 x^19 +  75.35 x^16 +  92.74 x^10 +  91.82
x^6
   x  0
   +  57.72 x^40 +  69.76 x^39 +  42.65 x^38 +  18.96 x^29 +  29.03 x^22 +  6.26 x^21  +  45.06
x^7
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
~~~~~PRINTING A/B~~~~~
A  =  68.33 x^43 +  50.18 x^40 +  6.26 x^39  +  45.06 x^38 +  42.65 x^29 +  32.06 x^28 +  42.02
x^22 +  57.72 x^21 +  29.03 x^7
B  =  89.30 x^42 +  71.53 x^38 +  92.74 x^37 +  73.50 x^31 +  86.13 x^24 +  63.37 x^19 +  24.41
x^16 +  55.92 x^10 +  6.59 x^6
   x  0.77 x^1
   +  50.18 x^40 +  -48.47 x^39    +  -25.90 x^38    +  -56.24 x^32    +  42.65 x^29 +  32.06
x^28 +  -65.90 x^25    +  42.02 x^22 +  57.72 x^21 +  -48.49 x^20    +  -18.68 x^17    +  -
42.78 x^11    +  23.99 x^7
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
~~~~~PRINTING A/B~~~~~
A  =  47.37 x^9
B  =  43.69 x^58
   x  0
   +  47.37 x^9
~~~~~~~~~~~~~~~~~~~~
```

```
~~~~~~~~~~~~~~~~~~~~
~~~~~PRINTING A/B~~~~~
A  =  26.14 x^59 +  14.48 x^18 +  5.75 x^9
B  =  43.41 x^58 +  35.66 x^32 +  13.99 x^12
   x  0.60 x^1
   +  -21.47 x^33    +  14.48 x^18 +  -8.42 x^13 +  5.75 x^9
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
~~~~~PRINTING A/B~~~~~
A  =  90.92 x^59 +  69.76 x^54 +  42.65 x^25 +  18.96 x^18 +  14.48 x^9
B  =  35.66 x^58 +  24.41 x^55 +  91.82 x^38 +  75.35 x^32 +  63.12 x^12
   x  2.55 x^1
   +  -62.24 x^56    +  69.76 x^54 +  -234.10 x^39   +  -192.12 x^33   +  42.65 x^25 +  18.96
x^18 +  -160.93 x^13   +  14.48 x^9
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
~~~~~PRINTING A/B~~~~~
A  =  69.76 x^59 +  29.03 x^58 +  6.26 x^54  +  45.06 x^25 +  42.65 x^18 +  18.96 x^9  +  57.72
B  =  75.35 x^58 +  89.30 x^55 +  6.59 x^38  +  24.41 x^32 +  91.82 x^12 +  63.37 x^5  +  92.74
x^3
   x  0.93 x^1   +  0.39
   +  -82.68 x^56    +  -34.41 x^55    +  6.26 x^54  +  -6.10 x^39 +  -2.54 x^38 +  -22.60 x^33
+  -9.40 x^32 +  45.06 x^25 +  42.65 x^18 +  -85.01 x^13    +  -35.38 x^12    +  18.96 x^9  +
-58.67 x^6 +  -24.42 x^5 +  -85.86 x^4 +  -35.73 x^3 +  57.72
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
~~~~~PRINTING A/B~~~~~
A  =  6.26 x^59  +  42.02 x^58 +  57.72 x^54 +  68.33 x^48 +  32.06 x^32 +  29.03 x^25 +  45.06
x^18 +  42.65 x^9  +  50.18
B  =  73.50 x^61 +  24.41 x^58 +  63.37 x^55 +  92.74 x^38 +  89.30 x^32 +  6.59 x^12  +  86.13
x^8  +  71.53 x^5  +  55.92 x^3
   x  0
   +  6.26 x^59  +  42.02 x^58 +  57.72 x^54 +  68.33 x^48 +  32.06 x^32 +  29.03 x^25 +  45.06
x^18 +  42.65 x^9  +  50.18
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
~~~~~PRINTING A/B~~~~~
A  =  47.37 x^53
B  =  43.69 x^48
   x  1.08 x^5
   +  0
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
~~~~~PRINTING A/B~~~~~
A  =  5.75 x^53  +  14.48 x^38 +  26.14 x^14
B  =  35.66 x^62 +  43.41 x^48 +  13.99 x^15
   x  0
   +  5.75 x^53  +  14.48 x^38 +  26.14 x^14
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
~~~~~PRINTING A/B~~~~~
A  =  14.48 x^53 +  42.65 x^51 +  18.96 x^38 +  69.76 x^20 +  90.92 x^14
B  =  75.35 x^62 +  91.82 x^53 +  35.66 x^48 +  24.41 x^23 +  63.12 x^15
   x  0
   +  14.48 x^53 +  42.65 x^51 +  18.96 x^38 +  69.76 x^20 +  90.92 x^14
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
~~~~~PRINTING A/B~~~~~
A  =  18.96 x^53 +  45.06 x^51 +  57.72 x^50 +  42.65 x^38 +  6.26 x^20  +  69.76 x^14 +  29.03
x^2
B  =  24.41 x^62 +  63.37 x^54 +  6.59 x^53  +  75.35 x^48 +  89.30 x^23 +  91.82 x^15 +  92.74
x^8
   x  0
   +  18.96 x^53 +  45.06 x^51 +  57.72 x^50 +  42.65 x^38 +  6.26 x^20  +  69.76 x^14 +  29.03
x^2
```

```
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
~~~~~PRINTING A/B~~~~~
A  =  42.65 x^53 +  29.03 x^51 +  50.18 x^50 +  45.06 x^38 +  32.06 x^36 +  68.33 x^33 +  57.72
x^20 +  6.26 x^14  +  42.02 x^2
B  =  89.30 x^62 +  71.53 x^54 +  92.74 x^53 +  24.41 x^48 +  63.37 x^23 +  86.13 x^16 +  6.59
x^15  +  73.50 x^11 +  55.92 x^8
   x  0
   +  42.65 x^53 +  29.03 x^51 +  50.18 x^50 +  45.06 x^38 +  32.06 x^36 +  68.33 x^33 +  57.72
x^20 +  6.26 x^14  +  42.02 x^2
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
~~~~~PRINTING A/B~~~~~
A  =  47.37 x^17
B  =  43.69 x^38
   x  0
   +  47.37 x^17
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
~~~~~PRINTING A/B~~~~~
A  =  14.48 x^38 +  26.14 x^21 +  5.75 x^17
B  =  35.66 x^52 +  43.41 x^38 +  13.99 x^30
   x  0
   +  14.48 x^38 +  26.14 x^21 +  5.75 x^17
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
~~~~~PRINTING A/B~~~~~
A  =  69.76 x^66 +  42.65 x^49 +  18.96 x^38 +  90.92 x^21 +  14.48 x^17
B  =  91.82 x^57 +  24.41 x^55 +  75.35 x^52 +  35.66 x^38 +  63.12 x^30
   x  0.76 x^9   +  -0.20 x^7  +  0.05 x^5   +  -0.62 x^4  +  -0.01 x^3  +  0.33 x^2   +  0.00
x^1   +  -0.13
   +  46.89 x^56 +  4.30 x^55  +  -24.98 x^54    +  -0.29 x^53 +  9.96 x^52  +  42.65 x^49 +  -
27.09 x^47    +  7.20 x^45  +  -1.92 x^43 +  22.24 x^42 +  0.51 x^41  +  -11.82 x^40    +  -
48.09 x^39    +  23.68 x^38 +  12.75 x^37 +  -3.39 x^35 +  39.36 x^34 +  0.90 x^33  +  -20.93
x^32    +  -0.24 x^31 +  8.34 x^30  +  90.92 x^21 +  14.48 x^17
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
~~~~~PRINTING A/B~~~~~
A  =  6.26 x^66  +  45.06 x^49 +  57.72 x^43 +  42.65 x^38 +  29.03 x^22 +  69.76 x^21 +  18.96
x^17
B  =  6.59 x^57  +  89.30 x^55 +  24.41 x^52 +  63.37 x^50 +  92.74 x^39 +  75.35 x^38 +  91.82
x^30
   x  0.95 x^9   +  -12.87 x^7 +  174.34 x^5 +  -3.52 x^4  +  -2362.40 x^3   +  86.18 x^2  +
32012.50 x^1   +  -1689.77
   +  -2858718.25 x^56   +  197518.38 x^55 +  -1880.67 x^54  +  -631705.31 x^53    +  35785.63
x^52  +  -2028730.75 x^51   +  107085.76 x^50 +  45.06 x^49 +  -88.05 x^48    +  -71.54 x^47
+  1193.14 x^46   +  969.39 x^45    +  -16168.00 x^44 +  -12752.19 x^43 +  219354.48 x^42 +
170011.98 x^41 +  -2975334.00 x^40   +  -2255475.75 x^39   +  127364.27 x^38 +  1181.26 x^37
+  -16006.99 x^35 +  322.88 x^34    +  216907.72 x^33 +  -7912.35 x^32  +  -2939275.75 x^31   +
155148.52 x^30 +  29.03 x^22 +  69.76 x^21 +  18.96 x^17
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
~~~~~PRINTING A/B~~~~~
A  =  57.72 x^66 +  68.33 x^58 +  29.03 x^49 +  50.18 x^43 +  45.06 x^38 +  32.06 x^28 +  42.02
x^22 +  6.26 x^21  +  42.65 x^17
B  =  86.13 x^62 +  92.74 x^57 +  63.37 x^55 +  89.30 x^52 +  71.53 x^50 +  73.50 x^46 +  55.92
x^39 +  24.41 x^38 +  6.59 x^30
   x  0.67 x^4
   +  -62.15 x^61    +  -42.47 x^59    +  68.33 x^58 +  -59.84 x^56    +  -47.93 x^54    +  -
49.25 x^50    +  29.03 x^49 +  12.71 x^43 +  -16.36 x^42    +  45.06 x^38 +  -4.42 x^34 +
32.06 x^28 +  42.02 x^22 +  6.26 x^21  +  42.65 x^17
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
~~~~~PRINTING A/B~~~~~
A  =  47.37 x^61
```

```
B  =  43.69 x^28
   x  1.08 x^33
   +  0
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
~~~~~PRINTING A/B~~~~~
A  =  14.48 x^78 +  5.75 x^61  +  26.14 x^22
B  =  13.99 x^90 +  43.41 x^28 +  35.66 x^2
   x  0
   +  14.48 x^78 +  5.75 x^61  +  26.14 x^22
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
~~~~~PRINTING A/B~~~~~
A  =  18.96 x^78 +  14.48 x^61 +  90.92 x^22 +  42.65 x^15 +  69.76 x^14
B  =  24.41 x^91 +  63.12 x^90 +  91.82 x^47 +  35.66 x^28 +  75.35 x^2
   x  0
   +  18.96 x^78 +  14.48 x^61 +  90.92 x^22 +  42.65 x^15 +  69.76 x^14
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
~~~~~PRINTING A/B~~~~~
A  =  42.65 x^78 +  29.03 x^68 +  18.96 x^61 +  57.72 x^42 +  69.76 x^22 +  45.06 x^15 +  6.26
x^14
B  =  89.30 x^91 +  91.82 x^90 +  6.59 x^47  +  75.35 x^28 +  92.74 x^8  +  63.37 x^6  +  24.41
x^2
   x  0
   +  42.65 x^78 +  29.03 x^68 +  18.96 x^61 +  57.72 x^42 +  69.76 x^22 +  45.06 x^15 +  6.26
x^14
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~
~~~~~PRINTING A/B~~~~~
A  =  68.33 x^88 +  45.06 x^78 +  42.02 x^68 +  42.65 x^61 +  32.06 x^56 +  50.18 x^42 +  6.26
x^22  +  29.03 x^15 +  57.72 x^14
B  =  63.37 x^91 +  6.59 x^90  +  73.50 x^81 +  92.74 x^47 +  86.13 x^44 +  24.41 x^28 +  55.92
x^8  +  71.53 x^6  +  89.30 x^2
   x  0
   +  68.33 x^88 +  45.06 x^78 +  42.02 x^68 +  42.65 x^61 +  32.06 x^56 +  50.18 x^42 +  6.26
x^22  +  29.03 x^15 +  57.72 x^14
~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
TESTING COMPLETED
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Time taken in ms is 2
```