Lab 3

Name: Pattabhiram Karthik Jonnalagadda

Lab section: B02

Course: ENCM 335

UCID: 30247339

Assignment: Lab 3

Exercise A:

code:

```
// ENCM 335 Fall 2025 Lab 3 Exercise A

#include <math.h>
// The above directive will read many function prototypes, including ...
//
//     double sin(double x);
//
// for the sine function.  It's assumed that the units for the argument
// value are radians, not degrees.

// Note to Linux users: You might need to build the executable like this ...
//   gcc -Wall lab3exA.c -lm
// ... in order to link in the sin function.  Mac and Cygwin users won't
// have to put -lm at the end of the command.

#include <stdio.h>

// For now, just trust that the following #define directive properly
// sets up a useful constant.  How it works will be explained
// later in the course.
#define PI 3.14159265358979323846

double deg2rad(double degrees);

int main(void)
{
    double row;

    printf("degrees");
    // printf(" THIS IS WHERE +0, +1, +2, ETC., SHOULD BE\n");
```

```
    // printf(" %+7d %+7d %+7d %+7d %+7d %+7d %+7d %+7d %+7d %+7d\n",
    //        0, 1, 2, 3, 4, 5, 6, 7, 8, 9 );
    for (int col = 0; col <= 9; col++)
    {
        printf(" %+7d", col);
    }
    printf("\n");

    for (row = 0; row <= 80; row += 10)
    {
        printf("%7.0f ", row);
        // printf(" %7.4f %7.4f %7.4f %7.4f %7.4f %7.4f %7.4f %7.4f %7.4f
%7.4f\n",
        //        sin(deg2rad((row))),sin(deg2rad(row+1)),
sin(deg2rad(row+2)), sin(deg2rad(row+3)), sin(deg2rad(row+4)),
sin(deg2rad(row+5)), sin(deg2rad(row+6)), sin(deg2rad(row+7)),
sin(deg2rad(row+8)), sin(deg2rad(row+9)) );

        for (int col = 0; col <= 9; col++)
        {
            printf("%7.4f ", sin(deg2rad(row + col)));
        }

        printf("\n");
    }
    return 0;
}

double deg2rad(double degrees)
{
    return (PI / 180.0) * degrees;
}
```

output:

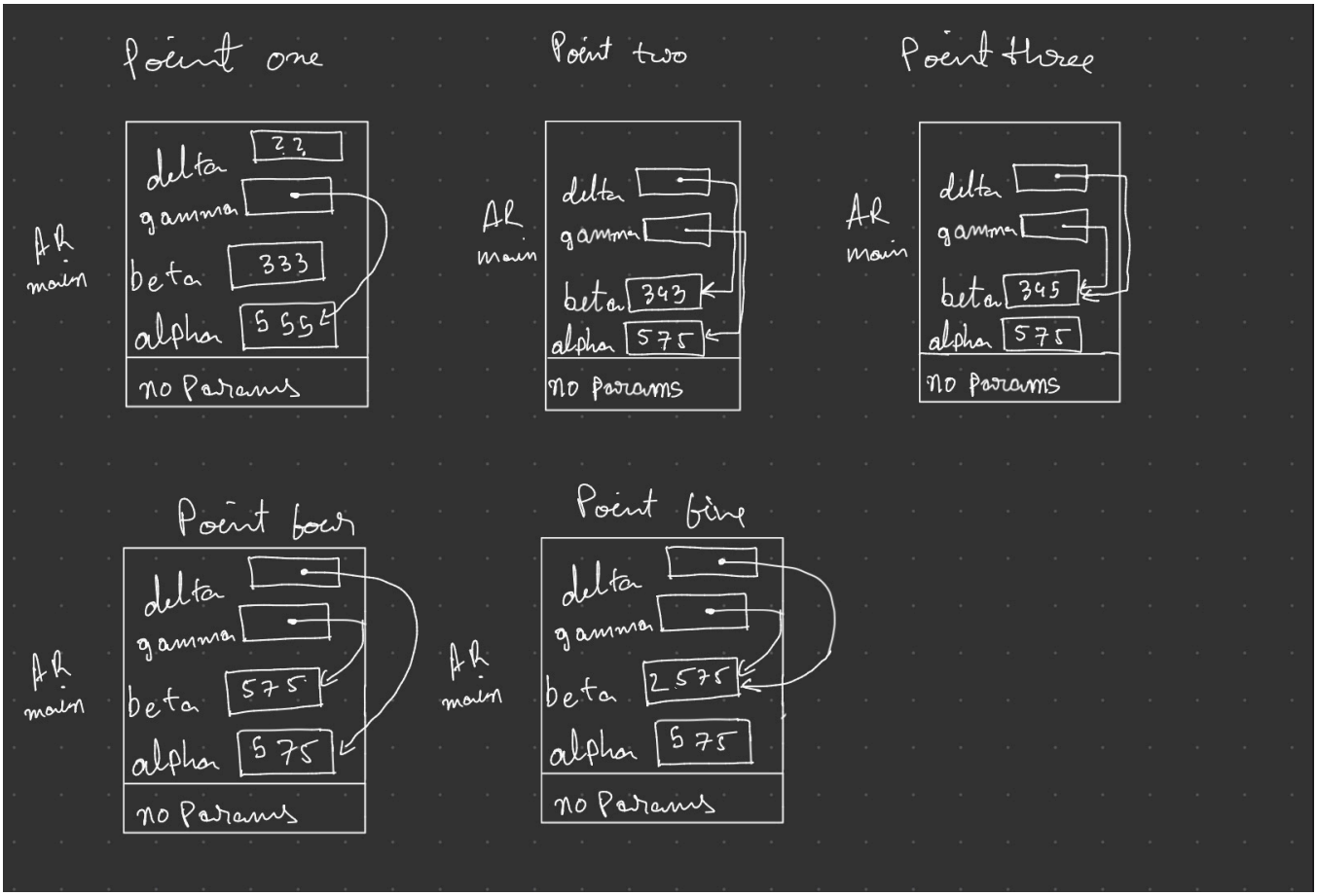
```
❯ ls
avg1.c   c2        lab3exA.c   lab3exC-partI      lab3exC-partIV.c
avg2.c   D.out     lab3exB     lab3exC-partI.c    lab3exD.c
c1       lab3exA   lab3exB.c   lab3exC-partIII.c  use-scanf.c
❯ gcc -Wall lab3exA.c -o A
❯ ./A
degrees      +0      +1      +2      +3      +4      +5      +6      +7      +8      +9
      0  0.0000  0.0175  0.0349  0.0523  0.0698  0.0872  0.1045  0.1219  0.1392  0.1564
     10  0.1736  0.1908  0.2079  0.2250  0.2419  0.2588  0.2756  0.2924  0.3090  0.3256
     20  0.3420  0.3584  0.3746  0.3907  0.4067  0.4226  0.4384  0.4540  0.4695  0.4848
     30  0.5000  0.5150  0.5299  0.5446  0.5592  0.5736  0.5878  0.6018  0.6157  0.6293
     40  0.6428  0.6561  0.6691  0.6820  0.6947  0.7071  0.7193  0.7314  0.7431  0.7547
     50  0.7660  0.7771  0.7880  0.7986  0.8090  0.8192  0.8290  0.8387  0.8480  0.8572
     60  0.8660  0.8746  0.8829  0.8910  0.8988  0.9063  0.9135  0.9205  0.9272  0.9336
     70  0.9397  0.9455  0.9511  0.9563  0.9613  0.9659  0.9703  0.9744  0.9781  0.9816
     80  0.9848  0.9877  0.9903  0.9925  0.9945  0.9962  0.9976  0.9986  0.9994  0.9998

~/desktop/ENCM 335/labs/lab3 on main !2 ?3 ····· at 12:54:02 AM
❯
```

Exercise B:

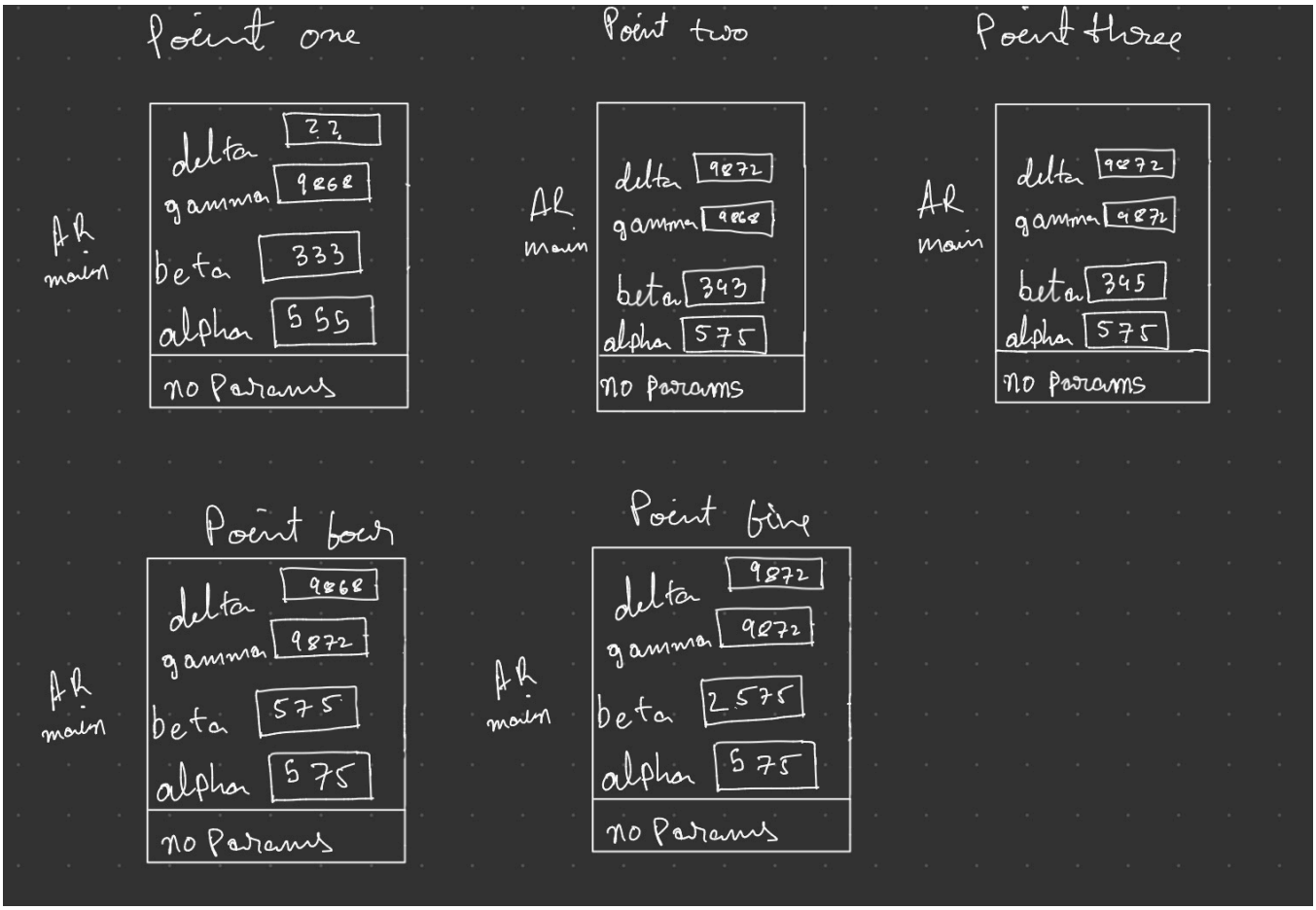
Variable	Address
ga	0x100eac000
gb	0x100eac004
la	0x16ef5ad78
lb	0x16ef5ad74

Exercise C:

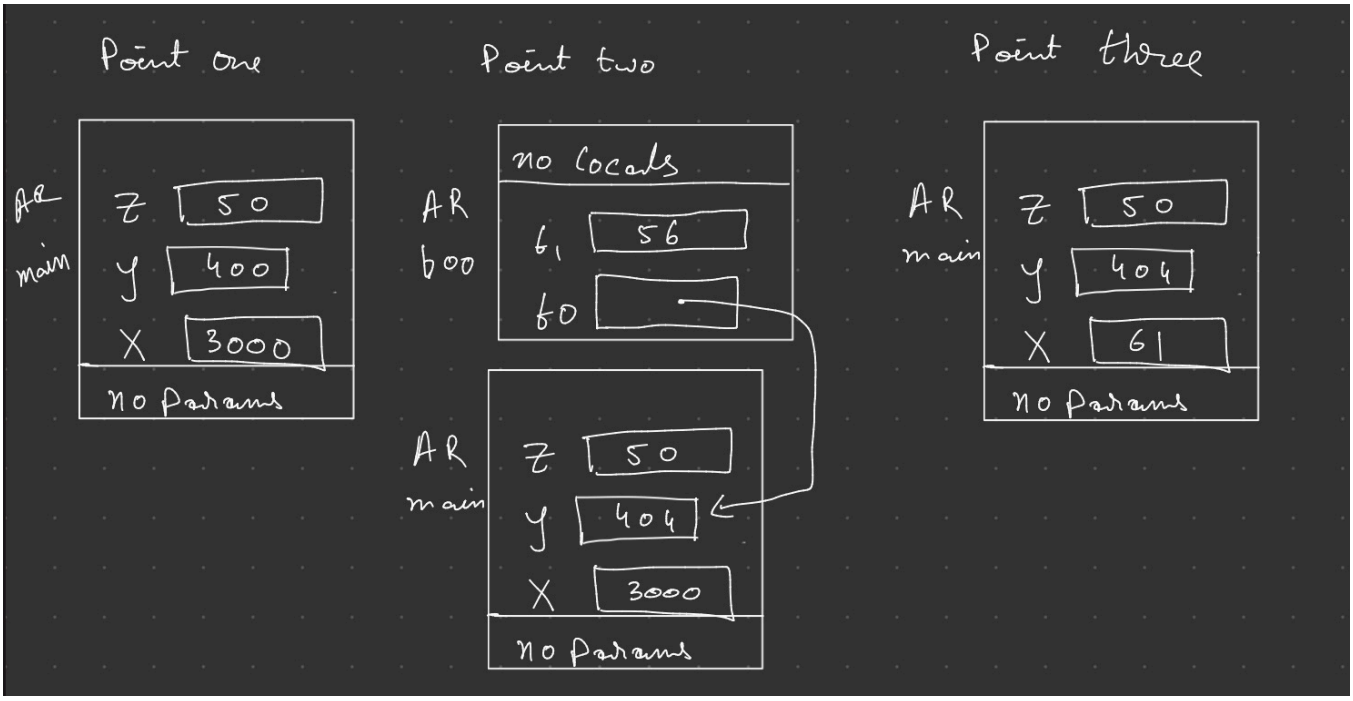
Part I:



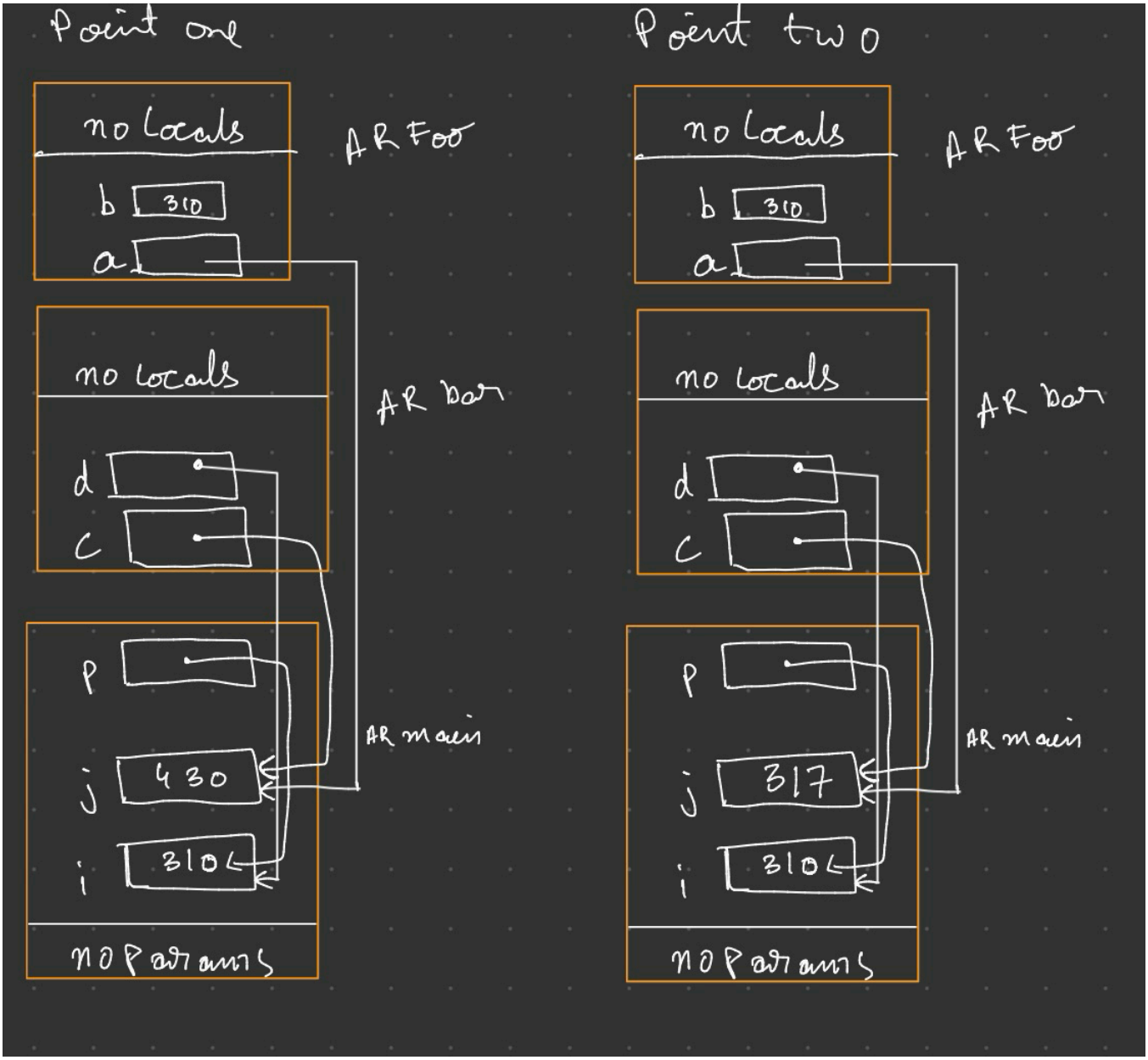
Part II:



Part III:


Point one
Point two
Point three
AR
main
z 50
y 400
X 3000
no params
AR
boo
no locals
b1 56
b0
AR
main
z 50
y 404
X 3000
no params
AR
main
z 50
y 404
X 61
no params

Part IV:



Exercise D:

Code:

```
//  lab2exD.c
//  ENCM 335 Fall 2025 Lab 3 Exercise D

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

void to_dhms(int total_s, int *d, int *h, int *min, int *s);
//
// Converts a number of seconds into days, hours, minutes, and seconds.
```

```c
// For example, 86403s is 1d, 0h, 0min, 3s.
// REQUIRES:
//   total_s >= 0.
//   Pointer parameters all point to appropriate variables.
// PROMISES:
//   *d contains number of days in conversion.
//   *h contains number of hours in conversion.
//   *min contains number of minutes in conversion.
//   *s contains number of seconds in conversion.

int main(void)
{
  int seconds_in, days, hours, minutes, seconds, scan_count;

  printf("Enter a number of seconds that is >= 0: ");
  scan_count = scanf("%d", &seconds_in);
  if (scan_count != 1)
  {
    printf("Unable to convert your input to an int.\n");
    exit(1);
  }
  if (seconds_in < 0)
  {
    printf("%d s is out of range!\n", seconds_in);
    exit(1);
  }

  printf("Doing conversion for input of %d s ... \n", seconds_in);

  // MAKE A CALL TO to_dhms HERE.
  to_dhms(seconds_in, &days, &hours, &minutes, &seconds);

  printf("That is %d day(s), %d hours(s), %d minute(s), %d second(s).\n",
         days, hours, minutes, seconds);

  return 0;
}

// WRITE A DEFINITION FOR to_dhms HERE.
// Hint: / for integer division and % for integer remainder are useful here.
// Another hint: There are 86400 seconds in a day.

void to_dhms(int total_s, int *d, int *h, int *min, int *s)
```

```
{

  int r = 0;
  *d = total_s / 86400;
  r = total_s % 86400;
  *h = r / (60 * 60);
  r = r % (60 * 60);
  *min = r / 60;
  r = r % 60;
  *s = r;
}
```

Output:

```
❯ gcc -Wall lab3exD.c -o D
❯ ./D
Enter a number of seconds that is >= 0: 86392
Doing conversion for input of 86392 s ...
That is 0 day(s), 23 hours(s), 59 minute(s), 52 second(s).
❯ ./D
Enter a number of seconds that is >= 0: 86408
Doing conversion for input of 86408 s ...
That is 1 day(s), 0 hours(s), 0 minute(s), 8 second(s).
❯ ./D
Enter a number of seconds that is >= 0: 309539
Doing conversion for input of 309539 s ...
That is 3 day(s), 13 hours(s), 58 minute(s), 59 second(s).
```

Exercise E:

q1) Yes, Both programs work the same way and give same result for same sample.

q2) With bad input, the programs behave **differently**:

- avg1.c fails (infinite loop).
- avg2.c safely terminates with an error message, because it explicitly checks for scanf returning 0.

#encm335