Date : 12/01/2026

PPL WEEK5 SUBMISSION

Name: Davasam Karthikeya   Section: AIMLB   Reg No: 230962326

1. Write a program in CUDA to add two vectors of length N using

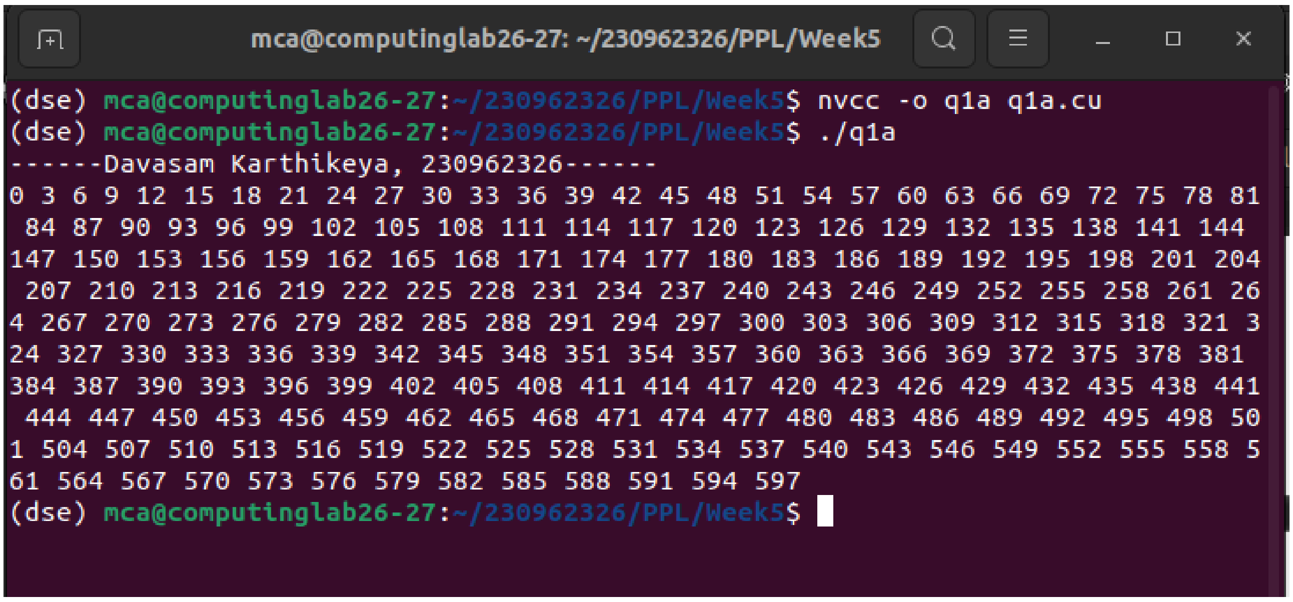
a) block size as N

Code:-

```
#include <cuda.h>
#include <stdio.h>

#define N 200

__global__ void add(int *a, int *b, int *c){
    int tid = blockIdx.x;
    c[tid] = a[tid] + b[tid];
}

int main(){
    int a[N], b[N], c[N];
    for(int i=0; i < N; i++){
        a[i] = i;
        b[i] = 2*i;
    }
    int *dA, *dB, *dC;

    int size = sizeof(int);

    printf("------Davasam Karthikeya, 230962326------\n");

    cudaMalloc((void **)&dA, N * size);
    cudaMalloc((void **)&dB, N * size);
    cudaMalloc((void **)&dC, N * size);

    cudaMemcpy(dA, &a[0], N * size, cudaMemcpyHostToDevice);
    cudaMemcpy(dB, &b[0], N * size, cudaMemcpyHostToDevice);

    add<<<N, 1>>>(dA, dB, dC);

    cudaMemcpy(c, dC, N * size, cudaMemcpyDeviceToHost);

    for(int i =0; i < N; i++)printf("%d ", c[i]);
    printf("\n");

    cudaFree(dA);
    cudaFree(dB);
    cudaFree(dB);

    return 0;
}
```

Date : 12/01/2026

Output:-



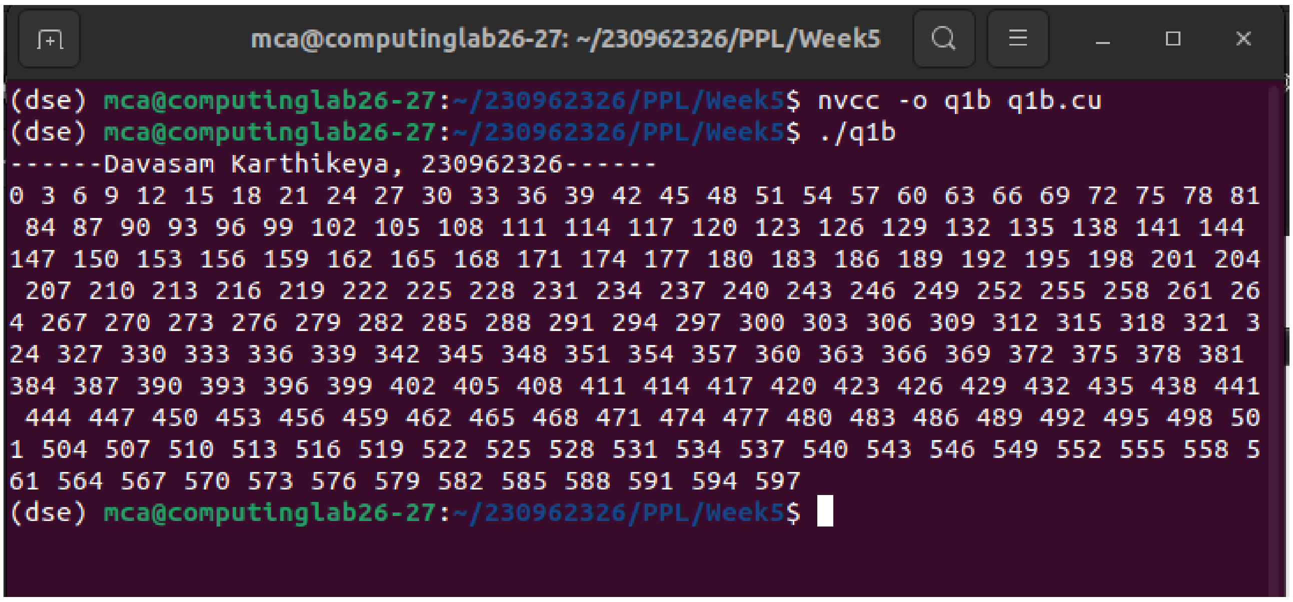
```
(dse) mca@computinglab26-27:~/230962326/PPL/Week5$ nvcc -o q1a q1a.cu
(dse) mca@computinglab26-27:~/230962326/PPL/Week5$ ./q1a
------Davasam Karthikeya, 230962326------
0 3 6 9 12 15 18 21 24 27 30 33 36 39 42 45 48 51 54 57 60 63 66 69 72 75 78 81
 84 87 90 93 96 99 102 105 108 111 114 117 120 123 126 129 132 135 138 141 144
147 150 153 156 159 162 165 168 171 174 177 180 183 186 189 192 195 198 201 204
 207 210 213 216 219 222 225 228 231 234 237 240 243 246 249 252 255 258 261 26
4 267 270 273 276 279 282 285 288 291 294 297 300 303 306 309 312 315 318 321 3
24 327 330 333 336 339 342 345 348 351 354 357 360 363 366 369 372 375 378 381
384 387 390 393 396 399 402 405 408 411 414 417 420 423 426 429 432 435 438 441
 444 447 450 453 456 459 462 465 468 471 474 477 480 483 486 489 492 495 498 50
1 504 507 510 513 516 519 522 525 528 531 534 537 540 543 546 549 552 555 558 5
61 564 567 570 573 576 579 582 585 588 591 594 597
(dse) mca@computinglab26-27:~/230962326/PPL/Week5$
```

b) N threads

Code:-

```
#include <cuda.h>
#include <stdio.h>

#define N 200

__global__ void add(int *a, int *b, int *c){
    int tid = blockDim.x * blockIdx.x + threadIdx.x;
    c[tid] = a[tid] + b[tid];
}

int main(){
    int a[N], b[N], c[N];
    for(int i=0; i < N; i++){
        a[i] = i;
        b[i] = 2*i;
    }
    int *dA, *dB, *dC;

    int size = sizeof(int);

    printf("------Davasam Karthikeya, 230962326------\n");

    cudaMalloc((void **)&dA, N * size);
    cudaMalloc((void **)&dB, N * size);
    cudaMalloc((void **)&dC, N * size);

    cudaMemcpy(dA, &a[0], N * size, cudaMemcpyHostToDevice);
    cudaMemcpy(dB, &b[0], N * size, cudaMemcpyHostToDevice);

    add<<<1,N>>>(dA, dB, dC);

    cudaMemcpy(c, dC, N * size, cudaMemcpyDeviceToHost);
    for(int i =0; i < N; i++)printf("%d ", c[i]);
    printf("\n");

    cudaFree(dA);
    cudaFree(dB);
    cudaFree(dB);

    return 0;
}
```

Output:-



```
(dse) mca@computinglab26-27:~/230962326/PPL/Week5$ nvcc -o q1b q1b.cu
(dse) mca@computinglab26-27:~/230962326/PPL/Week5$ ./q1b
------Davasam Karthikeya, 230962326------
0 3 6 9 12 15 18 21 24 27 30 33 36 39 42 45 48 51 54 57 60 63 66 69 72 75 78 81
 84 87 90 93 96 99 102 105 108 111 114 117 120 123 126 129 132 135 138 141 144
147 150 153 156 159 162 165 168 171 174 177 180 183 186 189 192 195 198 201 204
 207 210 213 216 219 222 225 228 231 234 237 240 243 246 249 252 255 258 261 26
4 267 270 273 276 279 282 285 288 291 294 297 300 303 306 309 312 315 318 321 3
24 327 330 333 336 339 342 345 348 351 354 357 360 363 366 369 372 375 378 381
384 387 390 393 396 399 402 405 408 411 414 417 420 423 426 429 432 435 438 441
 444 447 450 453 456 459 462 465 468 471 474 477 480 483 486 489 492 495 498 50
1 504 507 510 513 516 519 522 525 528 531 534 537 540 543 546 549 552 555 558 5
61 564 567 570 573 576 579 582 585 588 591 594 597
(dse) mca@computinglab26-27:~/230962326/PPL/Week5$
```

2. Implement a CUDA program to add two vectors of length N by keeping the number of threads per block as 256 (constant) and vary the number of blocks to handle N elements.

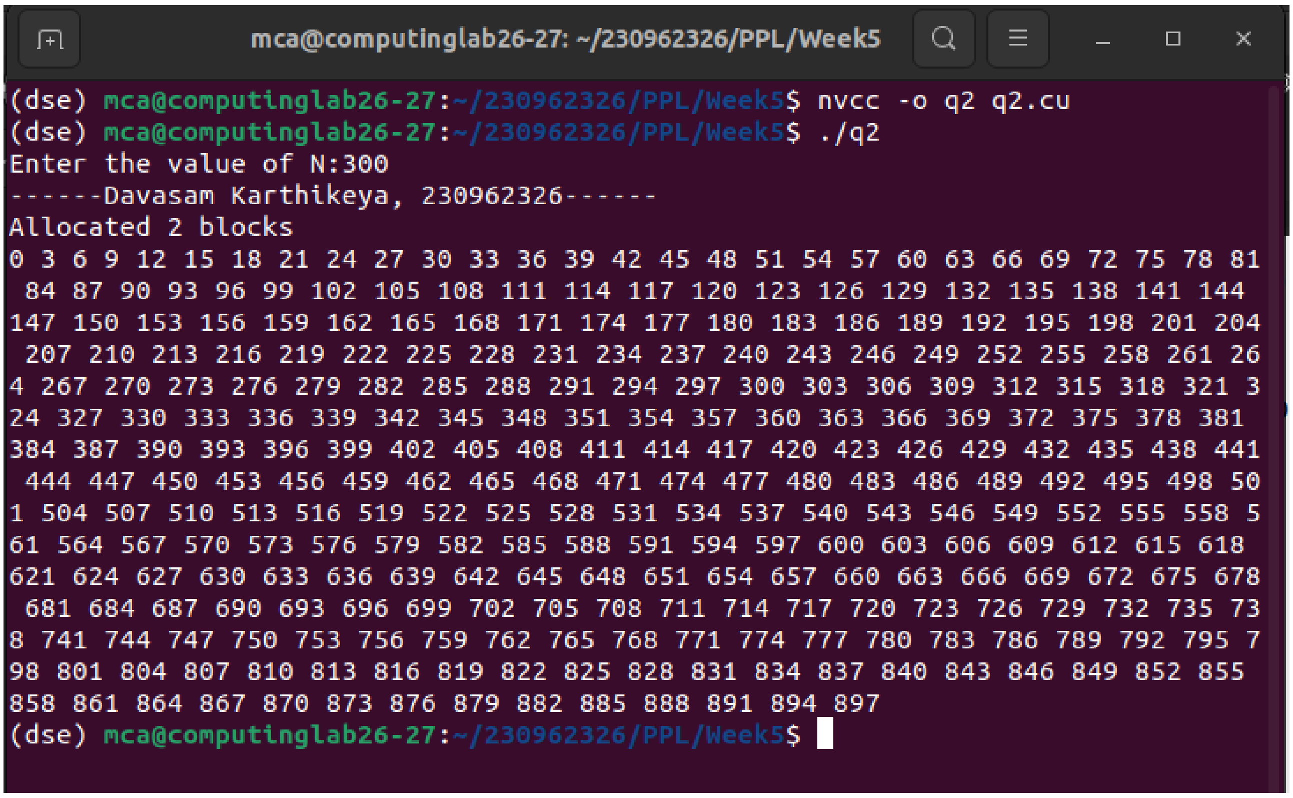
Code:-

```
#include <cuda.h>
#include <stdio.h>

__global__ void add(int N, int *a, int *b, int *c){
    int tid = blockDim.x * blockIdx.x + threadIdx.x;
    if(tid >= N)return;

    c[tid] = a[tid] + b[tid];
}

int main(){
    int N;
    printf("Enter the value of N:");
    scanf("%d", &N);

    int a[N], b[N], c[N];
    int *dA, *dB, *dC;
    for(int i = 0; i < N; i++){
        a[i] = i;
        b[i] = 2*i;
```

Date : 12/01/2026

```
    }

    int size = sizeof(int);

    printf("------Davasam Karthikeya, 230962326------\n");

    cudaMalloc((void **)&dA, N * size);
    cudaMalloc((void **)&dB, N * size);
    cudaMalloc((void **)&dC, N * size);

    cudaMemcpy(dA, &a[0], N * size, cudaMemcpyHostToDevice);
    cudaMemcpy(dB, &b[0], N * size, cudaMemcpyHostToDevice);

    int blocks = (N-1)/256 + 1;
    printf("Allocated %d blocks\n", blocks);

    add<<<blocks, 256>>>(N, dA, dB, dC);

    cudaMemcpy(c, dC, N * size, cudaMemcpyDeviceToHost);

    for(int i =0; i < N; i++)printf("%d ", c[i]);
    printf("\n");

    cudaFree(dA);
    cudaFree(dB);
    cudaFree(dB);

    return 0;
}
```

Output:-



```
(dse) mca@computinglab26-27:~/230962326/PPL/Week5$ nvcc -o q2 q2.cu
(dse) mca@computinglab26-27:~/230962326/PPL/Week5$ ./q2
Enter the value of N:300
------Davasam Karthikeya, 230962326------
Allocated 2 blocks
0 3 6 9 12 15 18 21 24 27 30 33 36 39 42 45 48 51 54 57 60 63 66 69 72 75 78 81
 84 87 90 93 96 99 102 105 108 111 114 117 120 123 126 129 132 135 138 141 144
147 150 153 156 159 162 165 168 171 174 177 180 183 186 189 192 195 198 201 204
 207 210 213 216 219 222 225 228 231 234 237 240 243 246 249 252 255 258 261 26
4 267 270 273 276 279 282 285 288 291 294 297 300 303 306 309 312 315 318 321 3
24 327 330 333 336 339 342 345 348 351 354 357 360 363 366 369 372 375 378 381
384 387 390 393 396 399 402 405 408 411 414 417 420 423 426 429 432 435 438 441
 444 447 450 453 456 459 462 465 468 471 474 477 480 483 486 489 492 495 498 50
1 504 507 510 513 516 519 522 525 528 531 534 537 540 543 546 549 552 555 558 5
61 564 567 570 573 576 579 582 585 588 591 594 597 600 603 606 609 612 615 618
621 624 627 630 633 636 639 642 645 648 651 654 657 660 663 666 669 672 675 678
 681 684 687 690 693 696 699 702 705 708 711 714 717 720 723 726 729 732 735 73
8 741 744 747 750 753 756 759 762 765 768 771 774 777 780 783 786 789 792 795 7
98 801 804 807 810 813 816 819 822 825 828 831 834 837 840 843 846 849 852 855
858 861 864 867 870 873 876 879 882 885 888 891 894 897
(dse) mca@computinglab26-27:~/230962326/PPL/Week5$
```

Date : 12/01/2026

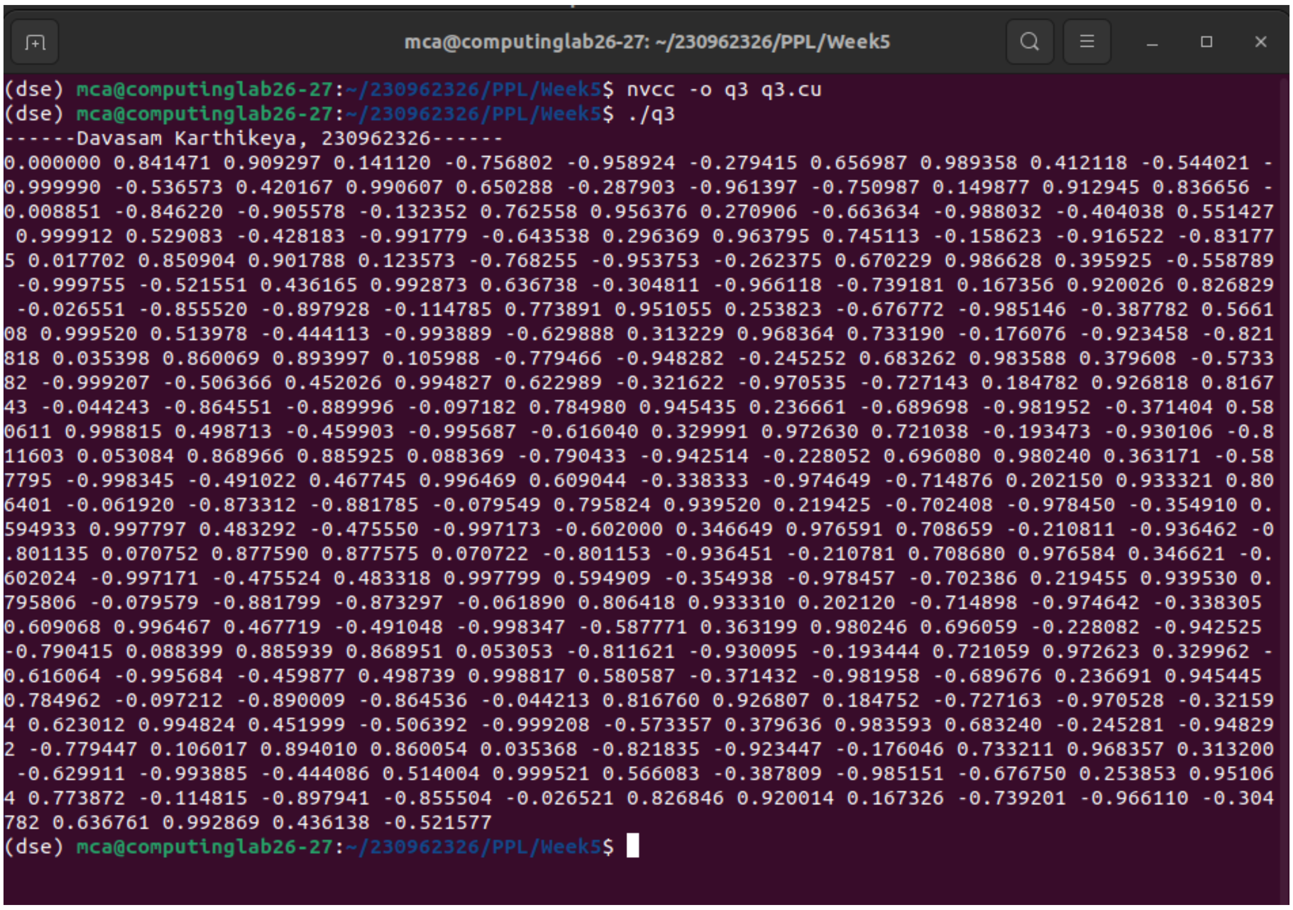
3. Write a program in CUDA to process a 1D array containing angles in radians to generate sine of the angles in the output array. Use appropriate function.

Code:-

```
#include <cuda.h>
#include <stdio.h>

#define N 300

__global__ void findSin(float *a, float *b){
    int tid = blockDim.x * blockIdx.x + threadIdx.x;

    if(tid >= N)return;
    b[tid] = sinf(a[tid]);
}

int main(){
    float a[N], b[N];
    for(int i = 0; i < N; i++)a[i]=i;
    float *dA, *dB;

    int size = sizeof(float);

    printf("------Davasam Karthikeya, 230962326------\n");

    cudaMalloc((void **)&dA, N * size);
    cudaMalloc((void **)&dB, N * size);

    cudaMemcpy(dA, &a[0], N * size, cudaMemcpyHostToDevice);

    findSin<<<(N-1)/256 + 1, 256>>>(dA, dB);

    cudaMemcpy(b, dB, N * size, cudaMemcpyDeviceToHost);

    for(int i =0; i < N; i++)printf("%f ", b[i]);
    printf("\n");

    cudaFree(dA);
    cudaFree(dB);

    return 0;
}
```

Date : 12/01/2026

Output:-



```
(dse) mca@computinglab26-27:~/230962326/PPL/Week5$ nvcc -o q3 q3.cu
(dse) mca@computinglab26-27:~/230962326/PPL/Week5$ ./q3
------Davasam Karthikeya, 230962326------
0.000000 0.841471 0.909297 0.141120 -0.756802 -0.958924 -0.279415 0.656987 0.989358 0.412118 -0.544021 -
0.999990 -0.536573 0.420167 0.990607 0.650288 -0.287903 -0.961397 -0.750987 0.149877 0.912945 0.836656 -
0.008851 -0.846220 -0.905578 -0.132352 0.762558 0.956376 0.270906 -0.663634 -0.988032 -0.404038 0.551427
 0.999912 0.529083 -0.428183 -0.991779 -0.643538 0.296369 0.963795 0.745113 -0.158623 -0.916522 -0.83177
5 0.017702 0.850904 0.901788 0.123573 -0.768255 -0.953753 -0.262375 0.670229 0.986628 0.395925 -0.558789
 -0.999755 -0.521551 0.436165 0.992873 0.636738 -0.304811 -0.966118 -0.739181 0.167356 0.920026 0.826829
 -0.026551 -0.855520 -0.897928 -0.114785 0.773891 0.951055 0.253823 -0.676772 -0.985146 -0.387782 0.5661
08 0.999520 0.513978 -0.444113 -0.993889 -0.629888 0.313229 0.968364 0.733190 -0.176076 -0.923458 -0.821
818 0.035398 0.860069 0.893997 0.105988 -0.779466 -0.948282 -0.245252 0.683262 0.983588 0.379608 -0.5733
82 -0.999207 -0.506366 0.452026 0.994827 0.622989 -0.321622 -0.970535 -0.727143 0.184782 0.926818 0.8167
43 -0.044243 -0.864551 -0.889996 -0.097182 0.784980 0.945435 0.236661 -0.689698 -0.981952 -0.371404 0.58
0611 0.998815 0.498713 -0.459903 -0.995687 -0.616040 0.329991 0.972630 0.721038 -0.193473 -0.930106 -0.8
11603 0.053084 0.868966 0.885925 0.088369 -0.790433 -0.942514 -0.228052 0.696080 0.980240 0.363171 -0.58
7795 -0.998345 -0.491022 0.467745 0.996469 0.609044 -0.338333 -0.974649 -0.714876 0.202150 0.933321 0.80
6401 -0.061920 -0.873312 -0.881785 -0.079549 0.795824 0.939520 0.219425 -0.702408 -0.978450 -0.354910 0.
594933 0.997797 0.483292 -0.475550 -0.997173 -0.602000 0.346649 0.976591 0.708659 -0.210811 -0.936462 -0
.801135 0.070752 0.877590 0.877575 0.070722 -0.801153 -0.936451 -0.210781 0.708680 0.976584 0.346621 -0.
602024 -0.997171 -0.475524 0.483318 0.997799 0.594909 -0.354938 -0.978457 -0.702386 0.219455 0.939530 0.
795806 -0.079579 -0.881799 -0.873297 -0.061890 0.806418 0.933310 0.202120 -0.714898 -0.974642 -0.338305
0.609068 0.996467 0.467719 -0.491048 -0.998347 -0.587771 0.363199 0.980246 0.696059 -0.228082 -0.942525
-0.790415 0.088399 0.885939 0.868951 0.053053 -0.811621 -0.930095 -0.193444 0.721059 0.972623 0.329962 -
0.616064 -0.995684 -0.459877 0.498739 0.998817 0.580587 -0.371432 -0.981958 -0.689676 0.236691 0.945445
0.784962 -0.097212 -0.890009 -0.864536 -0.044213 0.816760 0.926807 0.184752 -0.727163 -0.970528 -0.32159
4 0.623012 0.994824 0.451999 -0.506392 -0.999208 -0.573357 0.379636 0.983593 0.683240 -0.245281 -0.94829
2 -0.779447 0.106017 0.894010 0.860054 0.035368 -0.821835 -0.923447 -0.176046 0.733211 0.968357 0.313200
 -0.629911 -0.993885 -0.444086 0.514004 0.999521 0.566083 -0.387809 -0.985151 -0.676750 0.253853 0.95106
4 0.773872 -0.114815 -0.897941 -0.855504 -0.026521 0.826846 0.920014 0.167326 -0.739201 -0.966110 -0.304
782 0.636761 0.992869 0.436138 -0.521577
(dse) mca@computinglab26-27:~/230962326/PPL/Week5$
```