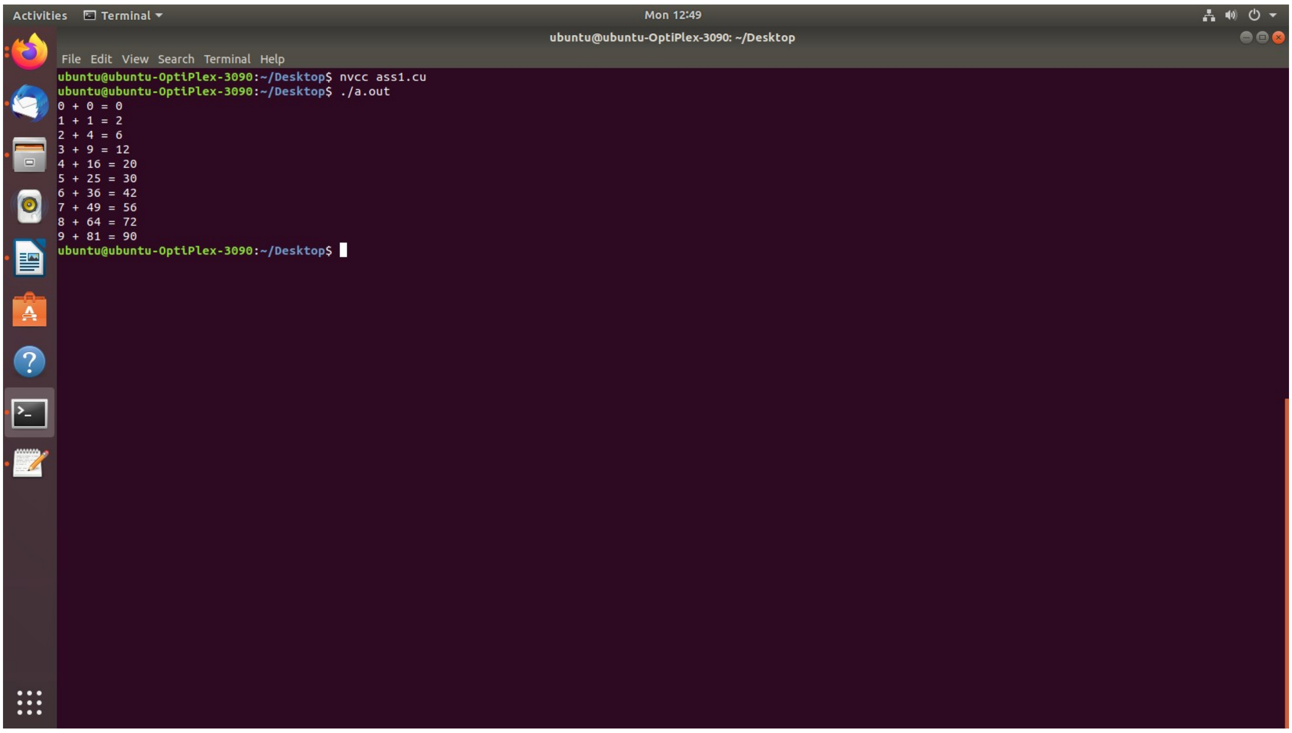
Cuda Program for Vector Addition

```
#include "stdio.h"
#include "math.h"
#define N 10
void add ( int *a, int *b, int *c )
{
              int tid = 0; // this is CPU zero, so we start at zero

              while  (tid < N)
              {
                      c[tid] = a[tid] + b[tid];
                       tid += 1;  // we have one CPU, so we increment by one
               }
}

int main( void )
{
             int a[N], b[N], c[N];

           // fill the arrays 'a' and 'b' on the CPU
           for (int i=0; i<N; i++)
           {
                      a [i] = i;
                      b[i] = i * i;
            }
         add( a, b, c );

       // display the results
           for (int i=0; i<N; i++)
         {
                printf( "%d + %d = %d\n", a[i], b[i], c[i] );
          }
  return 0;
}
```

```
ubuntu@ubuntu-OptiPlex-3090:~/Desktop$ nvcc ass1.cu
ubuntu@ubuntu-OptiPlex-3090:~/Desktop$ ./a.out
0 + 0 = 0
1 + 1 = 2
2 + 4 = 6
3 + 9 = 12
4 + 16 = 20
5 + 25 = 30
6 + 36 = 42
7 + 49 = 56
8 + 64 = 72
9 + 81 = 90
```



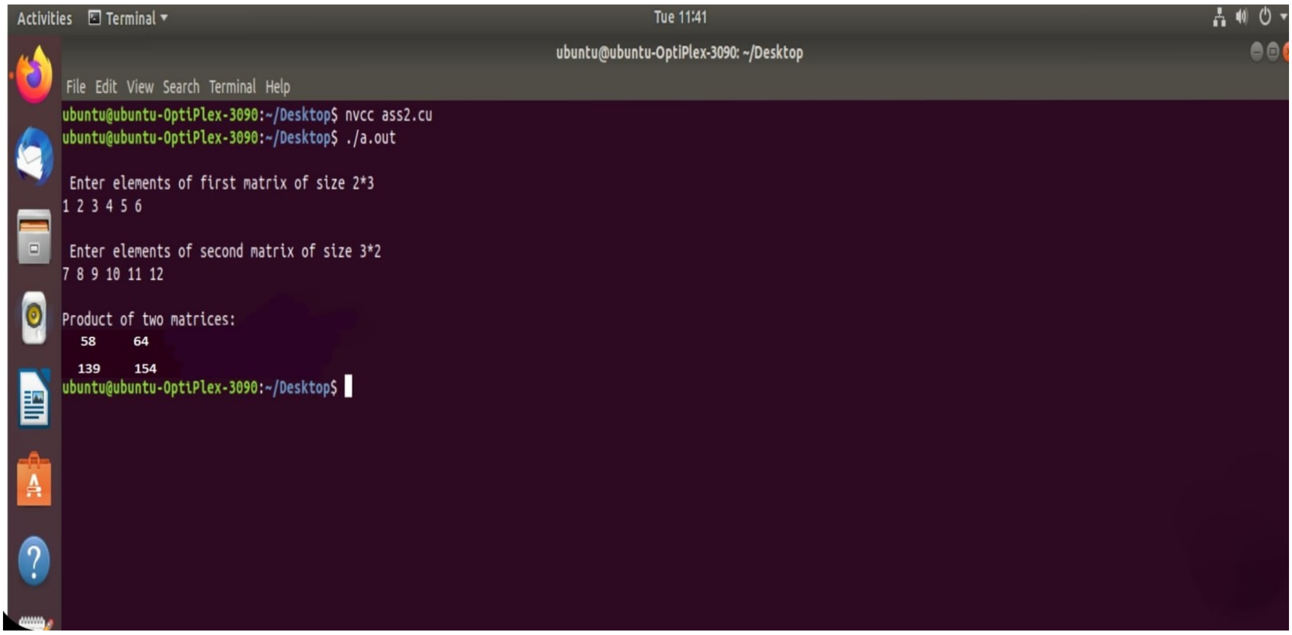
Activities
Terminal
Mon 12:49
ubuntu@ubuntu-OptiPlex-3090: ~/Desktop
File Edit View Search Terminal Help

```
ubuntu@ubuntu-OptiPlex-3090:~/Desktop$ nvcc ass1.cu
ubuntu@ubuntu-OptiPlex-3090:~/Desktop$ ./a.out
0 + 0 = 0
1 + 1 = 2
2 + 4 = 6
3 + 9 = 12
4 + 16 = 20
5 + 25 = 30
6 + 36 = 42
7 + 49 = 56
8 + 64 = 72
9 + 81 = 90
ubuntu@ubuntu-OptiPlex-3090:~/Desktop$
```

CUDA Code for matrix multiplication

```
 #include<stdio.h>
#include<cuda.h>
#define row1 2 /* Number of rows of first matrix */
#define col1 3 /* Number of columns of first matrix */
#define row2 3 /* Number of rows of second matrix */
#define col2 2 /* Number of columns of second matrix */

__global__ void matadd(int *l,int *m, int *n)
{
    int x=threadIdx.x;
    int y=threadIdx.y;

    int k;

n[col2*y+x]=0;
  for(k=0;k<col1;k++)
   {
    n[col2*y+x]=n[col2*y+x]+l[col1*y+k]*m[col2*k+x];
   }
}

int main()
{
    int a[row1][col1];
    int b[row2][col2];
    int c[row1][col2];
    int *d,*e,*f;
    int i,j;

    printf("\n Enter elements of first matrix of size 2*3\n");
    for(i=0;i<row1;i++)
    {
        for(j=0;j<col1;j++)
            {
                scanf("%d",&a[i][j]);
            }
    }
    printf("\n Enter elements of second matrix of size 3*2\n");
        for(i=0;i<row2;i++)
        {
            for(j=0;j<col2;j++)
                {
                    scanf("%d",&b[i][j]);
                }
        }

   cudaMalloc((void **)&d,row1*col1*sizeof(int));
    cudaMalloc((void **)&e,row2*col2*sizeof(int));
    cudaMalloc((void **)&f,row1*col2*sizeof(int));
  cudaMemcpy(d,a,row1*col1*sizeof(int),cudaMemcpyHostToDevice);
  cudaMemcpy(e,b,row2*col2*sizeof(int),cudaMemcpyHostToDevice);

dim3 threadBlock(col2,row1);
/* Here we are defining two dimensional Grid(collection of blocks) structure. Syntax is dim3
grid(no. of columns,no. of rows) */

    matadd<<<1,threadBlock>>>(d,e,f);

  cudaMemcpy(c,f,row1*col2*sizeof(int),cudaMemcpyDeviceToHost);

  printf("\nProduct of two matrices:\n ");
    for(i=0;i<row1;i++)
    {
        for(j=0;j<col2;j++)
        {
             printf("%d\t",c[i][j]);
        }
        printf("\n");
    }

    cudaFree(d);
    cudaFree(e);
    cudaFree(f);

    return 0;
}
```

output:-



Activities
Terminal ▾
Tue 11:41
ubuntu@ubuntu-OptiPlex-3090: ~/Desktop
File Edit View Search Terminal Help

```
ubuntu@ubuntu-OptiPlex-3090:~/Desktop$ nvcc ass2.cu
ubuntu@ubuntu-OptiPlex-3090:~/Desktop$ ./a.out

 Enter elements of first matrix of size 2*3
1 2 3 4 5 6

 Enter elements of second matrix of size 3*2
7 8 9 10 11 12

Product of two matrices:
 58	64	
 139	154	
ubuntu@ubuntu-OptiPlex-3090:~/Desktop$ 
```