

The following kernel function illustrates how these functions are used:

```
__kernel void square (__global int *input, __global int *output)
{
    int id = get_global_id(0);
    output[id] = input[id] * input[id];
}
```

The following figure shows the input and expected output:

input	4	5	2	7	1	0	9	3	1	2	7	8	5	6	1	6
output	16	25	4	49	1	0	81	9	1	4	49	64	25	36	1	36

Since the **kernel function** is a **data parallel function**, it is **executed for each work-item**.

When a work-item calls **get_global_id(0)** the request is **for the unique global workitem ID used to index the data**.

The example below, the **work-item instance** is **returned id=6** when it makes a call to **get_global_id(0)**. The **work-item** can then use the **id to index the data** to perform the operation.

get_global_id(0) → 6

input	4	5	2	7	1	0	9	3	1	2	7	8	5	6	1	6
output	16	25	4	49	1	0	81	9	1	4	49	64	25	36	1	36

If, for example, the kernel function is enqueued by indicating that the input will be divided into groups of eight elements, the results of the various work-item functions will appear as shown in the following image.

