

The issue is like passing variables through a function, where a pass-by-value creates a local copy of the variable in the function, while a pass-by-reference copies the pointer to the variable, essentially connecting the function directly to the variable.

```
hfani@bravo:~/lab09$ cc matrix_mul_row.c -o matrix_mul_row
hfani@bravo:~/lab09$ ./matrix_mul_row 2 3
Enter the matrix elements:
A[0,0] = 1
A[0,1] = 2
A[0,2] = 3

A[1,0] = 4
A[1,1] = 5
A[1,2] = 6

Enter a number:2
child0 PID: 4098201
2 * A[0,0] = 2
2 * A[0,1] = 4
2 * A[0,2] = 6
child1 PID: 4098202
2 * A[1,0] = 8
2 * A[1,1] = 10
2 * A[1,2] = 12
Final matrix elements:
A[0,0] = 1
A[0,1] = 2
A[0,2] = 3

A[1,0] = 4
A[1,1] = 5
A[1,2] = 6
```

From the example in the instructions, the yellow boxes (child programs) work on copies of the parent, so $2 * A[0,0]$ is saved in the matrix in the child, but the matrix in the parent is unaffected. Hence, $A[0,0]$ is still 1 in the parent matrix from start to finish.