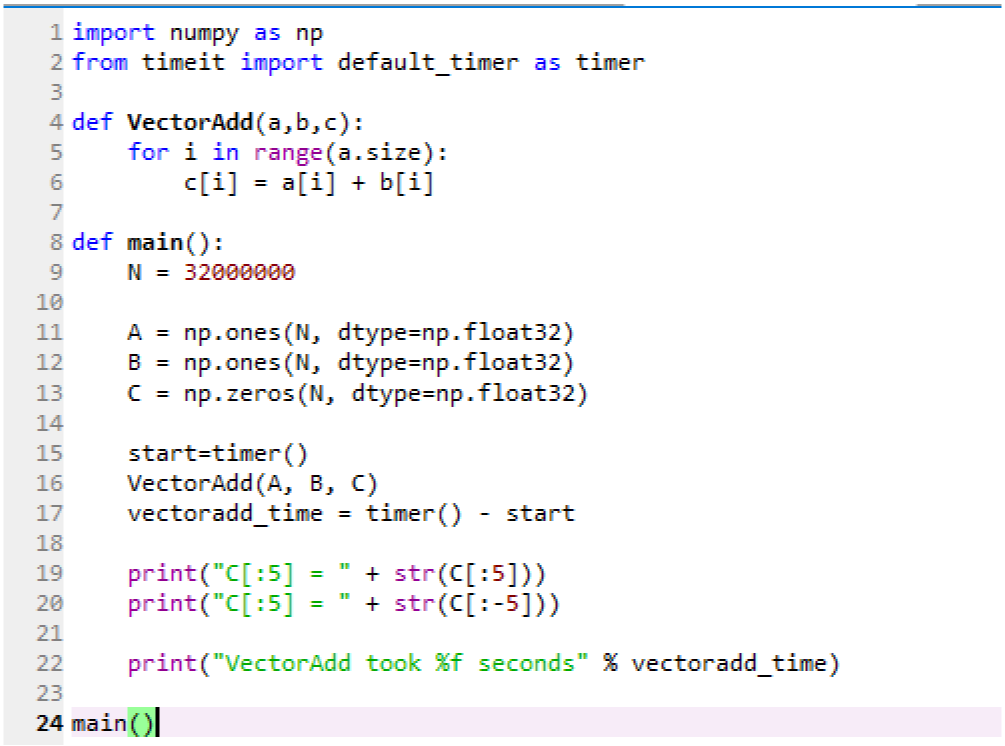
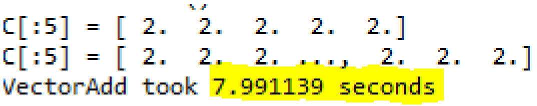
Code:



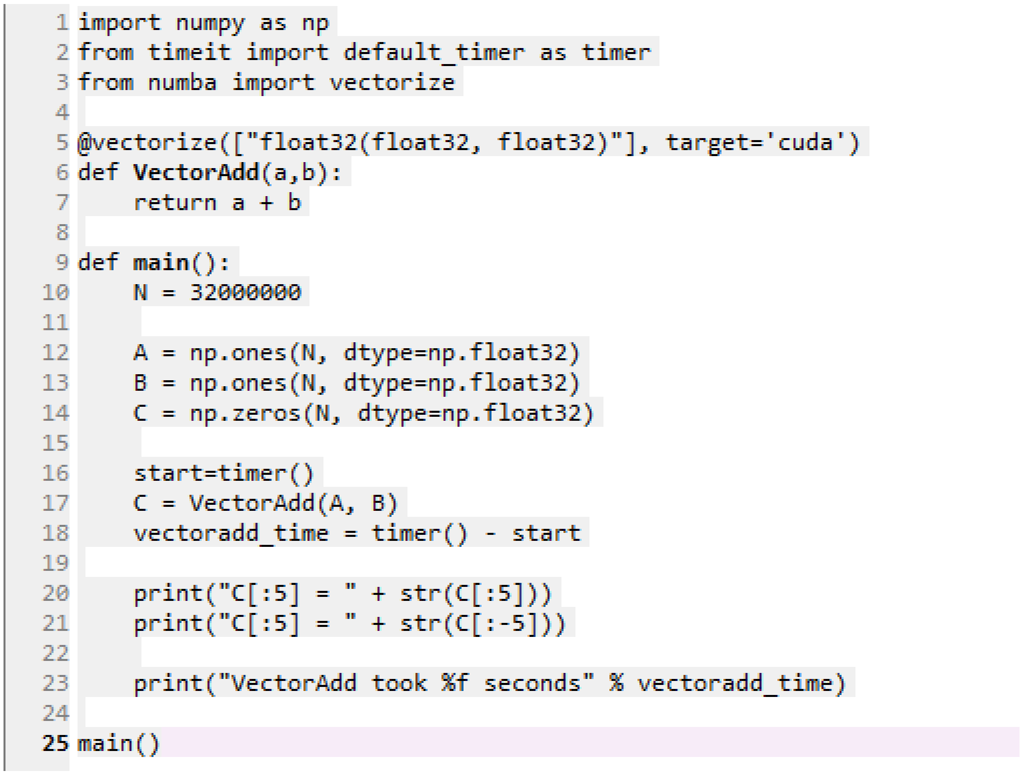
```python
import numpy as np
from timeit import default_timer as timer

def VectorAdd(a,b,c):
    for i in range(a.size):
        c[i] = a[i] + b[i]

def main():
    N = 32000000

    A = np.ones(N, dtype=np.float32)
    B = np.ones(N, dtype=np.float32)
    C = np.zeros(N, dtype=np.float32)

    start=timer()
    VectorAdd(A, B, C)
    vectoradd_time = timer() - start

    print("C[:5] = " + str(C[:5]))
    print("C[:5] = " + str(C[:-5]))

    print("VectorAdd took %f seconds" % vectoradd_time)

main()
```

Results:



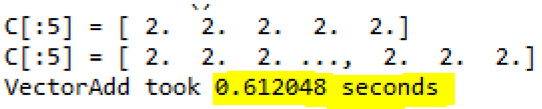
```
C[:5] = [ 2.  2.  2.  2.  2.]
C[:5] = [ 2.  2.  2. ...,  2.  2.  2.]
VectorAdd took 7.991139 seconds
```

New Code:



```
import numpy as np
from timeit import default_timer as timer
from numba import vectorize

@vectorize(["float32(float32, float32)"], target='cuda')
def VectorAdd(a,b):
    return a + b

def main():
    N = 32000000

    A = np.ones(N, dtype=np.float32)
    B = np.ones(N, dtype=np.float32)
    C = np.zeros(N, dtype=np.float32)

    start=timer()
    C = VectorAdd(A, B)
    vectoradd_time = timer() - start

    print("C[:5] = " + str(C[:5]))
    print("C[:5] = " + str(C[:-5]))

    print("VectorAdd took %f seconds" % vectoradd_time)

main()
```

Results:



```
C[:5] = [ 2.  2.  2.  2.  2.]
C[:5] = [ 2.  2.  2. ...,  2.  2.  2.]
VectorAdd took 0.612048 seconds
```

GPU - .612(!)

Vs

CPU – 7.99