2. a) I used the method described in lecture notes (code included in file integral.c, function simpson). It calculates the integral using Simpson's rule and then estimates the error. If error is bigger than a pre-set tolerance, this is repeated for a shorter section of the domain of integration. This is repeated until desired accuracy is achieved or maximum recursion dept is reached, after which the value of the integral is summed from integrals of smaller parts of the function.

I got value 0.6000000000 when calculating from 0 to 1e10 and setting maximum number of recursion levels to 80 and eps to 1e-10.

b) See next page for transformation. For transformed function, I hard-coded value f(t=0)=0 in, because the function is not defined at t=0. After changing function f to represent the function after transformation and running simpson with limits 0 to 1 and same eps and maximum recursion level I got the same value of 0.6000000000.

c) Gauss-Laguerre quadrature approximates integral of a function with a weighted sum of values within the domain of integration. Here weights and points in which values are calculated are determined from Laguerre polynomials. I implemented Gauss-Laguerre quadrature as described in Wikipedia (https://en.wikipedia.org/wiki/Gauss%E2%80%93Laguerre_quadrature) using Matlab which has a built-in implementation for Laguerre polynomials. My implementation can be found in file laguerreIntegral.m. With N=2, N=4 and N=8 I got values 0.7285, 0.5551 and 0.5961 respectively. Getting value 0.6000 seems to require that N is at least 16.

3. b) Romberg method estimates the value of an integral by using trapezium rule and then refining its value by using Richardson extrapolation. I implemented Romberg method following the code given in lecture notes (file romberg.c) and ran it with different functions and values of n. Outputs were following:

```
A) n=3:  0.5935251183
   n=8:  0.6022869520
   n=10: 0.6023310353
B) n=3:  0.6020615634
   n=8:  0.6023370697
   n=10: 0.6023373397
```

It therefore seems that the latter is somewhat more accurate, the correct value being about 0.60234.