3g. Trajectory Inversion

- A particle travels along the x-direction with velocity as a function of space:

$$v(x) = \frac{1}{\sqrt{2 + (\sin x)^4}}$$

- Find the particle position $x^* = x(t^*)$ when $t^* = 5$. To this purpose, consider

$$\frac{dx}{dt} = v(x) \quad \Longrightarrow \quad t(x) = \int_0^x \frac{dx}{v(x)}$$

  and invert the last relation to find x^* given t*, that is, solve $t(x^*) = t^*$ using Newton's method with a tolerance $\varepsilon = 10^{-6}$.

- Use Gaussian quadrature with N_g=3 Gaussian points to evaluate the integral and print, each time you compute the integral:
  i. the lower (x_{lo}) and upper ($x_{hi} \equiv x$) bounds as well as the the number of intervals `n_int=ceil(2*|x_lo - x_hi|)` used to compute the integral;
  ii. Make the computation as efficient as possible and print the cumulative number of function ($1/v(x)$) evaluation calls (`nfv += Ng*n_int`).

- Upload your code with i) the output inserted in the comments at the beginning of the file, ii) the required library function at the end, e.g.

```
// Name: First Name, Last name
// Date: 30 Oct 2025
//
// Code output:
// *****************************************************
// xlo = ..; xhi = ..; nint = ..; nfv = ..
// ...
// Particle position (t=5) = ..
// *****************************************************
#include ...
...
int main()
{
  // code here
}

double Time(double x){
  static int nfv = 0; // Cumulative number of function evaluations
  ...
  GaussianQuad(..);

  nfv += Ng*nint;
  cout << "xlo = "    << xlo  << "; xhi = " << x << "; nint = " << nint
       << "; nfv = " << nfv << endl;
  ...
}

double Inv_Velocity(double x){
...
}
<your library functions here…>
```