Ejercicio Tema 3 – PROA

4.

Deberia de ocupar 12 bytes debido a la alineación. Bool ocupa 1 byte. Float ocupa 4 bytes pero por alineación deben de haber 3 bytes previos sin usar. Unsigned short ocupa 2 bytes. Para que la estructura este alineada se usan 2 bytes posteriores. En total 12 bytes por instancia, 5 bytes malastados.
Modificando el orden de los atributos se podria almezenar en 8 bytes malgastando solo 1.
Demostrado en el código con un sizeof.

5.

```
unsigned search_optimizado(vector<int> &v, int x)
{
    unsigned pos = 0;
    v.push_back(x); //Añadir valor a buscar al final

    while(v[pos] != x)
        ++pos;  //Usar ++pos en lugar de pos++

    v.pop_back(); //Quitamos el elemento

    return pos;
}
```

Se consigue una mejora del 30%.

6.

Iteración	Cambio	Tiempo
0	Original	4.474s
1	De i++ a ++i	4.367s
2	De y = y +… a y += ...	4.380 s
3	Crear función MyPow	2.645 s

Se podría intentar desenrollar el bucle de eval pero no lo hago porque no esta garantizado que el tamaño del polinomio a evaluar sea mayor de 1 elemento. Habría que añadir algún if para evitar salirse del rango haciéndolo menos eficiente,

7.
a)

```
int i, j, k;
float x[N], y[N];

for(k = 0; k < ITER; ++k)
{
     x[0] = 0;

     for(i = 1; i < N-1; ++i)
     {
          x[i] += y[i];
     }

     x[N-1] = N-1;
}
```

b)

```
int i, j, k;
float a, x[N], y[N];

for(k = 0; k < ITER; k++)
{
     if (a == 0.0)
     {
          for(i = 0; i < N; i++)
          {
               x[i] = 0;
          }
     }
     else
     {
          for(i = 0; i < N; i++)
          {
               y[i] *= x[i];
          }
     }
}
```

c)

```
int i, j, k;
float x[N], y[N];
float a = 0.0;

for(i=0; i<N; i++)
{
     a += x[i] * y[i];
}
```