PRACTICAL - 08

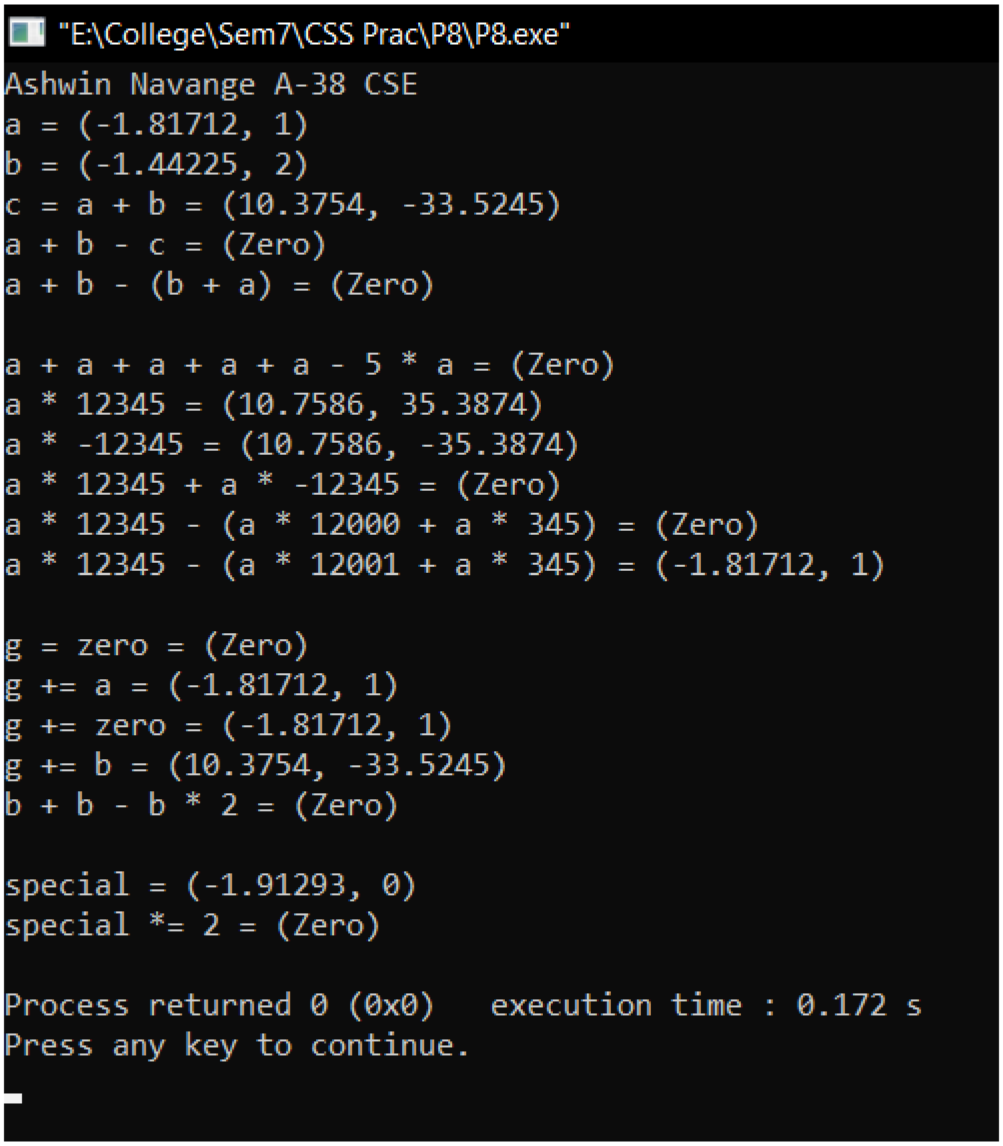
Code:

```
// Ashwin Navange A-38 CSE
#include <cmath>
#include <iostream>
using namespace std;
class EllipticPoint
{
   double m_x, m_y;
   static constexpr double ZeroThreshold = 1e20;
   static constexpr double B = 7;

   void Double() noexcept
   {
      if(IsZero())
      {
         return;
      }
      if(m_y == 0)
      {
         *this = EllipticPoint();
      }
      else
      {
         double L = (3 * m_x * m_x) / (2 * m_y);
         double newX = L * L -  2 * m_x;
         m_y = L * (m_x - newX) - m_y;
         m_x = newX;
      }
   }

public:
   friend std::ostream& operator<<(std::ostream&, const EllipticPoint&);
   constexpr EllipticPoint() noexcept : m_x(0), m_y(ZeroThreshold * 1.01) {}
   explicit EllipticPoint(double yCoordinate) noexcept
   {
      m_y = yCoordinate;
      m_x = cbrt(m_y * m_y - B);
   }
   bool IsZero() const noexcept
   {
      bool isNotZero =  abs(m_y) < ZeroThreshold;
      return !isNotZero;
   }
   EllipticPoint operator-() const noexcept
   {
      EllipticPoint negPt;
      negPt.m_x = m_x;
      negPt.m_y = -m_y;

      return negPt;
```

```
}
EllipticPoint& operator+=(const EllipticPoint& rhs) noexcept
{
   if(IsZero())
   {
      *this = rhs;
   }
   else if (rhs.IsZero())
   {
      // since rhs is zero this point does not need to be
      // modified
   }
   else
   {
      double L = (rhs.m_y - m_y) / (rhs.m_x - m_x);
      if(isfinite(L))
      {
         double newX = L * L - m_x - rhs.m_x;
         m_y = L * (m_x - newX) - m_y;
         m_x = newX;
      }
      else
      {
         if(signbit(m_y) != signbit(rhs.m_y))
         {
            *this = EllipticPoint();
         }
         else
         {
            Double();
         }
      }
   }

   return *this;
}
EllipticPoint& operator-=(const EllipticPoint& rhs) noexcept
{
   *this+= -rhs;
   return *this;
}
EllipticPoint& operator*=(int rhs) noexcept
{
   EllipticPoint r;
   EllipticPoint p = *this;

   if(rhs < 0)
   {
      rhs = -rhs;
      p = -p;
   }

   for (int i = 1; i <= rhs; i <<= 1)
```

```
        {
            if (i & rhs) r += p;
            p.Double();
        }

        *this = r;
        return *this;
    }
};
inline EllipticPoint operator+(EllipticPoint lhs, const EllipticPoint& rhs) noexcept
{
    lhs += rhs;
    return lhs;
}
inline EllipticPoint operator-(EllipticPoint lhs, const EllipticPoint& rhs) noexcept
{
    lhs += -rhs;
    return lhs;
}
inline EllipticPoint operator*(EllipticPoint lhs, const int rhs) noexcept
{
    lhs *= rhs;
    return lhs;
}
inline EllipticPoint operator*(const int lhs, EllipticPoint rhs) noexcept
{
    rhs *= lhs;
    return rhs;
}
ostream& operator<<(ostream& os, const EllipticPoint& pt)
{
    if(pt.IsZero()) cout << "(Zero)\n";
    else cout << "(" << pt.m_x << ", " << pt.m_y << ")\n";
    return os;
}

int main(void) {
    const EllipticPoint a(1), b(2);
    cout<<"Ashwin Navange A-38 CSE"<<endl;
    cout << "a = " << a;
    cout << "b = " << b;
    const EllipticPoint c = a + b;
    cout << "c = a + b = "       << c;
    cout << "a + b - c = "       << a + b - c;
    cout << "a + b - (b + a) = " << a + b - (b + a) << "\n";

    cout << "a + a + a + a + a - 5 * a = "         << a + a + a + a + a - 5 * a;
    cout << "a * 12345 = "                         << a * 12345;
    cout << "a * -12345 = "                        << a * -12345;
    cout << "a * 12345 + a * -12345 = "            << a * 12345 + a * -12345;
    cout << "a * 12345 - (a * 12000 + a * 345) = " << a * 12345 - (a * 12000 + a * 345);
    cout << "a * 12345 - (a * 12001 + a * 345) = " << a * 12345 - (a * 12000 + a * 344) << "\n";
    const EllipticPoint zero;
    EllipticPoint g;
    cout << "g = zero = "      << g;
    cout << "g += a = "        << (g+=a);
    cout << "g += zero = "     << (g+=zero);
    cout << "g += b = "        << (g+=b);
    cout << "b + b - b * 2 = " << (b + b - b * 2) << "\n";

    EllipticPoint special(0);  // the point where the curve crosses the x-axis
    cout << "special = "      << special; // this has the minimum possible value for x
    cout << "special *= 2 = " << (special*=2); // doubling it gives zero

    return 0;
}
```

Output:



```
"E:\College\Sem7\CSS Prac\P8\P8.exe"
Ashwin Navange A-38 CSE
a = (-1.81712, 1)
b = (-1.44225, 2)
c = a + b = (10.3754, -33.5245)
a + b - c = (Zero)
a + b - (b + a) = (Zero)

a + a + a + a + a - 5 * a = (Zero)
a * 12345 = (10.7586, 35.3874)
a * -12345 = (10.7586, -35.3874)
a * 12345 + a * -12345 = (Zero)
a * 12345 - (a * 12000 + a * 345) = (Zero)
a * 12345 - (a * 12001 + a * 345) = (-1.81712, 1)

g = zero = (Zero)
g += a = (-1.81712, 1)
g += zero = (-1.81712, 1)
g += b = (10.3754, -33.5245)
b + b - b * 2 = (Zero)

special = (-1.91293, 0)
special *= 2 = (Zero)

Process returned 0 (0x0)   execution time : 0.172 s
Press any key to continue.
```