```cpp
#include "Polygon.hpp"

std::set<Edge> Polygon::allEdges;

// Costruttori
Polygon::Polygon(std::vector<Point2D>& v)
{
    if(v.size() < 3)  throw std::invalid_argument("Error! Trying to create a polygon with less than three edges");
    myEdges.reserve(v.size());
    for(size_t i = 0; i < v.size()-1; i++)
    {
        myEdges.push_back(allEdges.emplace(v[i],v[i+1]).first);
    }
    myEdges.push_back(allEdges.emplace(v.back(),v[0]).first);
}
Polygon::Polygon(std::vector<Edge>& v)
{
    if(v.size() < 3)  throw std::invalid_argument("Error! Trying to create a polygon with less than three edges");
    myEdges.reserve(v.size());
    myEdges.push_back(allEdges.insert(v[0]).first);
    for(size_t i = 0; i < v.size(); i++)
    {
        if (!Edge::Connected(*myEdges.back(), v[i])) throw std::invalid_argument("Error! Trying to create a non connected polygon");
        myEdges.push_back(allEdges.insert(v[i]).first);
    }
}

// Distruttore
Polygon::~Polygon()
{

}

// Costruttore di copia
Polygon::Polygon(const Polygon& other) : myEdges(other.myEdges)
{
    std::cout << "Copio un Polygon\n";
}

Polygon& Polygon::operator=(const Polygon& other)
{
    std::cout << "Copio un Polygon tramite assignement operator\n";
    myEdges = other.myEdges;
    return *this;
}

// Metodi per l'accesso alle coordinate dei vertici
std::vector<Point2D> Polygon::Vertices()
{
```

```cpp
    std::vector<Point2D> v;
    v.reserve(myEdges.size());
    v.push_back(myEdges[0]->getA());
    v.push_back(myEdges[0]->getB());
    for(size_t i = 1; i < myEdges.size(); i++)
    {
        Point2D A = myEdges[i]->getA();
        if(v.back() == A) v.push_back(myEdges[i]->getB());
        else v.push_back(A);
    }
    return v;
}

// Metodo per l'accesso al numero dei lati
size_t Polygon::edgesNum()
{
    return myEdges.size();
}

// Metodo per l'accesso ai lati
std::vector<Edge> Polygon::Edges()
{
    std::vector<Edge> v;
    v.reserve(myEdges.size());
    for(size_t i = 0; i < myEdges.size(); i++)
    {
        v.push_back(*myEdges[i]);
    }
    return v;
}

// Metodo per il calcolo del perimetro
double Polygon::Perimeter()
{
    double p = 0.0;
    for(size_t i = 0; i < myEdges.size(); i++) p+=myEdges[i]->length();
    return p;
}

// Metodo per il caloclo dell'area
double Polygon::Area()
{
    std::vector<Point2D> v = Vertices();
    double a = 0;
    for(size_t i = 0; i < v.size(); ++i)
        {
                a += v[i].getX() * v[(i+1)%v.size()].getY();
                a -= v[i].getY() * v[(i+1)%v.size()].getX();
        }
        return 0.5*std::abs(a);
}

```