Copilot

Your idea of defining a `Point` struct with two floats (presumably for the x and y coordinates) and a `Polygon` struct with an unsigned integer for the number of vertices and a doubly linked list of `Point` structs is a good start. Here's a basic example of how you might define these structs in C:

```
typedef struct Point {
    float x;
    float y;
    struct Point* next;
    struct Point* prev;
} Point;

typedef struct Polygon {
    unsigned int nvertices;
    Point* vertices;
} Polygon;
```

In this example, `Point` is a doubly linked list node with `next` and `prev` pointers to connect to other points. The `Polygon` struct contains the number of vertices and a pointer to the first node in the doubly linked list of vertices.

Remember, in a doubly linked list, you can traverse in both directions but it uses extra memory to store the `prev` pointer. If you don't need to traverse in both directions, a singly linked list would use less memory.

Also, you'll need to write functions to add and delete nodes from your linked list, as well as to handle memory allocation and deallocation for your nodes.

This is a good exercise to understand data structures and memory management in C. Good luck with your project! 😊