Tutorial n°4 - Structures and pointers: application to lists

We are going to develop simple (and naïve) structures to handle 2D closed polygons. As we are going to see through this example, we do not necessarily need to handle arrays when allocating the memory. Actually, the corresponding data structure created will be a double chained list.

1 2D Point

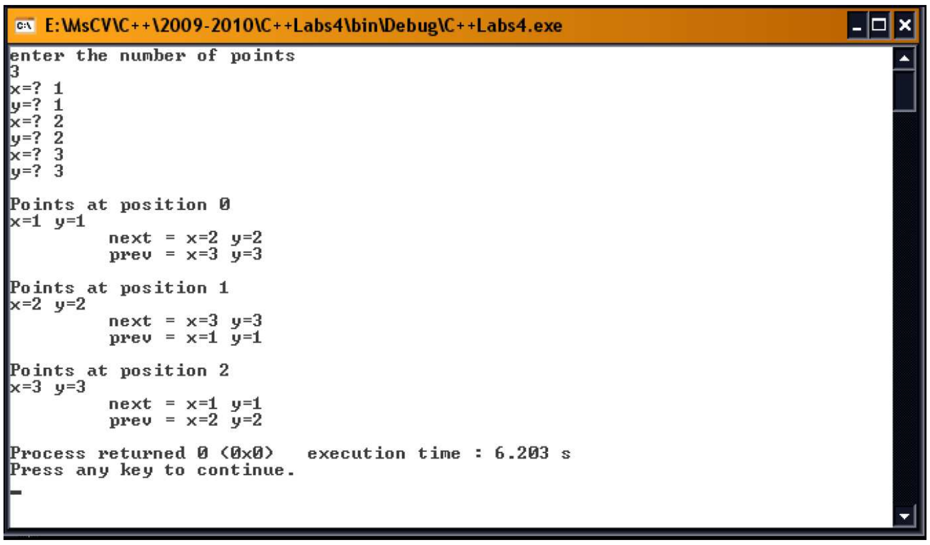
1. Declare and implement a structure *Point2d* to represent a $2D$ point which has to members x and y represented by floats.
2. Implement two functions $display(\ldots)$ that display on the screen the values of members x and y for a Point2d and for a pointer to a Point2d.
3. Declare and implement functions $BuildPoint(\ldots)$ that asks values of members x and y for a $2D$ point (by pointer and by reference).
4. Declare and initialize a "dummy" *Point2d* within the main function to test your results.

2 Polygon

To represent a polygon, we need to handle multiple $2D$ points. Additionally, we need to know the order of the points within the polygon. To do so, update the structure *Point2d* by adding two members *prev* and *next*, that are pointers to the predecessor and the successors of a point within a polygon.

1. Declare and implement a structure *Polygon2d* that contains a pointer to an initial 2D point called *start*.
2. Declare and implement a function $BuildPolygon(\ldots)$, that asks to the user the number of points of the polygon, then the coordinates of each point (within the order of the polygon), and then that creates the $2D$ points.
3. Declare and implement a function that displays the elements of a polygon. This function should also display the previous and next points within the polygon.

You should obtain the following result:


E:\MsCV\C++\2009-2010\C++Labs4\bin\Debug\C++Labs4.exe

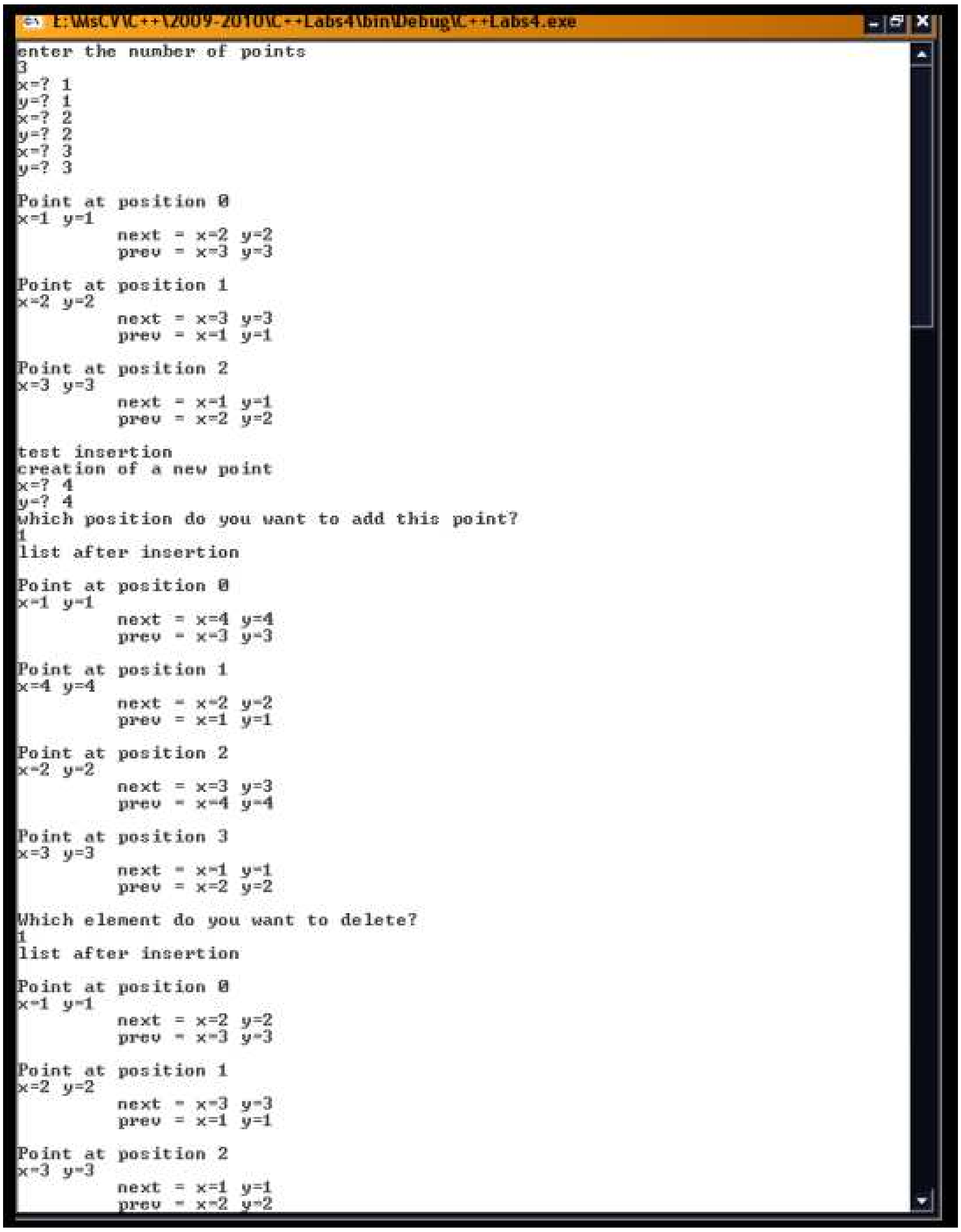
```
enter the number of points
3
x=? 1
y=? 1
x=? 2
y=? 2
x=? 3
y=? 3

Points at position 0
x=1 y=1
         next = x=2 y=2
         prev = x=3 y=3

Points at position 1
x=2 y=2
         next = x=3 y=3
         prev = x=1 y=1

Points at position 2
x=3 y=3
         next = x=1 y=1
         prev = x=2 y=2

Process returned 0 (0x0)   execution time : 6.203 s
Press any key to continue.
_
```

3 Insertion and deletion of elements

Declare and implement the following functions:

1. $GetElement(\dots)$ that returns a pointer to a $2D$ Point at position in a given polygon.
2. $GetPosition(\dots)$ that returns the position of a point within a given polygon.
3. $InsertAt(\dots)$ that inserts an element at a given position.
4. $DeleteAt(\dots)$, that deletes (if possible) an element at position I.

You should obtain these kinds of results:



```
E:\MsCV\C++\2009-2010\C++Labs4\bin\Debug\C++Labs4.exe
enter the number of points
3
x=? 1
y=? 1
x=? 2
y=? 2
x=? 3
y=? 3

Point at position 0
x=1 y=1
          next = x=2 y=2
          prev = x=3 y=3

Point at position 1
x=2 y=2
          next = x=3 y=3
          prev = x=1 y=1

Point at position 2
x=3 y=3
          next = x=1 y=1
          prev = x=2 y=2

test insertion
creation of a new point
x=? 4
y=? 4
which position do you want to add this point?
1
list after insertion

Point at position 0
x=1 y=1
          next = x=4 y=4
          prev = x=3 y=3

Point at position 1
x=4 y=4
          next = x=2 y=2
          prev = x=1 y=1

Point at position 2
x=2 y=2
          next = x=3 y=3
          prev = x=4 y=4

Point at position 3
x=3 y=3
          next = x=1 y=1
          prev = x=2 y=2

Which element do you want to delete?
1
list after insertion

Point at position 0
x=1 y=1
          next = x=2 y=2
          prev = x=3 y=3

Point at position 1
x=2 y=2
          next = x=3 y=3
          prev = x=1 y=1

Point at position 2
x=3 y=3
          next = x=1 y=1
          prev = x=2 y=2
```