University of Burgundy
Software Engineering
Tutorial No 4

SHAH Bhargav
DUDHAGARA AkshayKumar

4 November 2018

Exercise 1

2D Point
1. Declare and implement a structure point2D to represent a 2D point which has to member x and y represented by floats.
2. Implement two functions $Display(\cdots)$ 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 $BuildPoint(\cdots)$ 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.

Exercise 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 $BuildPolygon(\cdots)$,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.

Exercise 3

Insertion and deletion of elements
1. $GetElement(\cdots)$ that returns a pointer to a 2D Point at position in a given polygon.
2. $GetPosition(\cdots)$ that returns the position of a point within a given polygon.
3. $InsertAt(\cdots)$ that inserts an element at a given position.

4. *DeleteAt*$(\cdots)$ that deletes (if possible) an element at position I.

Listing 1: Structure and Pointer

```


****************************** Header file.h*******************************

#include <iostream>
using namespace std;

//                         Exercise 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
struct Point2d
{
    float x, y,prevx, prevy, nextx, nexty;
};

// 2.Implement two functions display(...) that display on the screen the values of members x and y
void display(Point2d a); // For a Point2d
void display(Point2d *a); // For a pointer to a Point2d

// 3.Declare function BuildPoint(...) that asks values of members x and y for a 2D point
void BuildPoint(Point2d *a); // By pointer
void BuildPoint ( Point2d &a); // By reference


//                         Exercise 2: Polygon

// 1.Declare a structure Polygon2d that contains a pointer to an initial 2D point called start
struct Polygon2d
{
    int n;
    Point2d *start;  // Pointer to an initial 2D point start
};

// 2.Declare and implement a function BuildPolygon(...)
void Buildneighbor(Point2d val[], int n, int i);
void Buildpolygon(Point2d val[], int n);
void displaypolygon(Point2d *ptr, int n);
void displayviapointer(Point2d *ptr);


//                         Exercise 3: Insertion and deletion of elements

// 3.InsertAt(...) that inserts an element at a given position
Point2d* Insertelement(Point2d val[], int in, int n);
// 4.DeleteAt(...)that deletes an element at position I
Point2d* delelement(Point2d val3[], int delptr, int n);


\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\


*********************Source file or function file**************************************


#include "labs4.h"

//                         Exercise 1: 2D Point

// 1.2:Implement two functions display(...) that display on the screen the values of members x and y
void display(Point2d m) // The values of members x and y for a Point2d
 {
```

```
62      cout << "Value of member X is " << m.x << " and Value of member Y is " << m.y <<
            endl;
63  }
64
65  void display(Point2d *m) // The values of members x and y for a pointer to a Point2d
66  {
67      cout << "Value of member X is " << m->x << " and Value of member Y is " << m->y <<
            endl;
68  }
69
70  // 1.3:Declare function BuildPoint(...) that asks values of members x and y for a 2D
        point
71  void BuildPoint( Point2d *m) // By pointer
72  {
73      cout <<"Type the value of member X ";
74      cin >> m->x;
75      cout <<"Type the value of member Y ";
76      cin >> m->y;
77  }
78
79  void BuildPoint ( Point2d &m) // By reference
80  {
81      cout <<"Type the value of member X ";
82      cin >> m.x;
83      cout <<"Type the value of member Y ";
84      cin >> m.y;
85  }
86
87  //                          Exercise 2: Polygon
88
89  // 2.2
90  // create a function for the value of prevx,prevy,nextx,nexty
91  void Buildneighbor(Point2d val[], int n, int i)
92  {
93      // set boundary for the value of i
94      if(i == 0)
95      {
96          val[i].prevx = val[n].x;
97          val[i].prevy = val[n].y;
98          val[i].nextx = val[1].x;
99          val[i].nexty = val[1].y;
100     }else
101     if(i == n)
102     {
103         val[i].prevx = val[n-1].x;
104         val[i].prevy = val[n-1].y;
105         val[i].nextx = val[0].x;
106         val[i].nexty = val[0].y;
107     }else
108     {
109         val[i].prevx = val[i-1].x;
110         val[i].prevy = val[i-1].y;
111         val[i].nextx = val[i+1].x;
112         val[i].nexty = val[i+1].y;
113     }
114
115     cout << endl;
116 }
117
118 void Buildpolygon(Point2d val[], int n)
119 {
120     Point2d *ptr1; // temp variable as a pointer
121
122     ptr1 = val;
123     for(int i = 0; i < n; i++)
124     {
125         BuildPoint(ptr1); // call a function for x and y
126         ptr1++;
127     }
```

```
128
129     ptr1 = val;
130     for(int i = 0; i < n; i++) // loop for polygon
131     {
132         Buildneighbor(ptr1, n-1, i);
133
134     }
135
136 }
137
138 // 2.3: Displays the elements of a polygon and also display the previous and next points
        within the polygon
139 void displayviapointer(Point2d *ptr)
140 {
141     cout << "Previous elements are: " <<"X = " << ptr->prevx << " and  Y= " << ptr->prevy
            << endl;
142     cout << "Entered Point is: "  <<"X = " << ptr->x << " and Y= " << ptr->y << endl;
143     cout << "Next elements are: "  <<"X = " << ptr->nextx << " and Y= " << ptr->nexty <<
            endl;
144     cout << endl;
145 }
146
147 void displaypolygon(Point2d *ptr, int n)
148 {
149     Point2d *disp = ptr;
150     cout << "Displaying the elements and their neighbors: " << endl;
151     for(int i = 0; i < n; i++)
152     {
153         cout <<"Point at position "<<i<<endl;
154         displayviapointer(disp);
155         disp++;
156     }
157 }
158
159
160 //                Exercise 3: Insertion and deletion of elements
161
162 // 3.3
163 Point2d* Insertelement(Point2d val[], int in, int n) // inserts an element at a given
        position
164 {
165     Point2d* val2 = new Point2d[n+1]; // assign a new list
166
167     for(int i = 0; i < n+1; i++)
168     {
169         if(i==n)
170             val2[i] = val[i-1];
171         else
172             val2[i] = val[i];
173     }
174
175     for(int i = n; i > in; i--)
176     {
177         val2[i].x = val2[i-1].x;
178         val2[i].y = val2[i-1].y;
179         val2[i].nextx = val2[i-1].nextx;
180         val2[i].nexty = val2[i-1].nexty;
181         val2[i].prevx = val2[i-1].prevx;
182         val2[i].prevy = val2[i-1].prevy;
183 //        (ptrIn + i)->x = (ptrIn + i - 1)->x;
184 //        (ptrIn + i)->y = (ptrIn + i - 1)->y;
185 //        (ptrIn + i)->nextx = (ptrIn + i - 1)->nextx;
186 //        (ptrIn + i)->nexty = (ptrIn + i - 1)->nexty;
187 //        (ptrIn + i)->prevx = (ptrIn + i - 1)->prevx;
188 //        (ptrIn + i)->prevy = (ptrIn + i - 1)->prevy;
189
190     }
191
192     Point2d* ptrIn = val2;
```

```

    BuildPoint((ptrIn+in));
    for(int i = 0; i < n+1; i++)
    {
        Buildneighbor(ptrIn, n, i); // update the element of x and y
    }
    return val2;
}

// 3.4
Point2d* delelement(Point2d val3[], int delptr, int n) // delete an element at position I
{
    Point2d* val31 = new Point2d[n-1];
    for(int i = 0; i < n-1; i++)
    {
        if(i >= delptr)
            val31[i] = val3[i+1];
        else
            val31[i] = val3[i];
    }

//    for(int i = delptr; i < n-1; i++)
//    {
//        val2[i].x = val2[i+1].x;
//        val2[i].y = val2[i+1].y;
//        val2[i].nextx = val2[i+1].nextx;
//        val2[i].nexty = val2[i+1].nexty;
//        val2[i].prevx = val2[i+1].prevx;
//        val2[i].prevy = val2[i+1].prevy;
//    }
    Point2d* ptrIn = val31;

    for(int i = 0; i < n-1; i++)
    {
        Buildneighbor(ptrIn, n-2, i);
    }
    return val31;
}

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\

*********************************main file.cpp******************************

#include "labs4.h"
#include <iostream>
using namespace std;

int main()
{
    cout << "LAB 4: Structures and pointers: application to lists" << endl;
    cout << "Submitted by Bhargav SHAH and Akshaykumar DUDHAGARA" << endl;
    cout << "Guided by Yohan Fougerolle " << endl;
    cout << endl;
    cout << endl;

    int n, a, in, delptr;

    //                        Exercise 1: 2D Point
    cout << "Exercise 1" << endl;
    Point2d m;
    Point2d *b=&m; // pointer is equal to the address of the object
    BuildPoint(b);

    // Display the values of members x and y for a Point2d
    cout<< endl;
    cout<<"The values of members x and y for a Point2d"<< endl;
    display(m);

    // Display the values of members x and y for a pointer to a Point2d
```

```
262     cout<< endl;
263     cout<<"The values of members x and y for a pointer to a Point2d"<< endl;
264     display(b);
265
266
267 //                          Exercise 2: Polygon
268     cout << endl;
269     cout << "Exercise 2" << endl;
270     cout << "Enter the number of points for a polygon(n): " << endl;
271     cin >> n;
272     cout << endl;
273
274     Point2d* val = new Point2d[n]; // assign a new list
275     Point2d *ptr;
276     ptr = val;
277
278     Buildpolygon(val, n);
279     displaypolygon(ptr, n); // display polygon
280     cout << endl;
281
282
283     //                      Exercise 3: Insertion and deletion of elements
284
285     cout << "Exercise 3" << endl;
286     cout << "Enter the value for which information is required(<=n): " << endl;
287     cin >> a;
288     displayviapointer(ptr+a-1);
289     cout << endl;
290
291     if(a > n)
292     {
293         cout << "WRONG data entered!!!" << endl;
294         exit(EXIT_FAILURE);
295     }
296
297     cout << "Enter the value at which you want to enter the element(<=n): " << endl;
298     cin >> in;
299
300     if(in > n)
301     {
302         cout << "WRONG data entered!!!" << endl;
303         exit(EXIT_FAILURE);
304     }
305     Point2d* newptr = Insertelement(val, in-1, n);
306     cout << endl;
307     n++;
308     displaypolygon(newptr, n);
309     cout << endl;
310
311     cout << "Enter the value at which you want to delete from the array(<=n): " << endl;
312     cin >> delptr;
313
314     if(delptr > n)
315     {
316         cout << "WRONG data entered!!!" << endl;
317         exit(EXIT_FAILURE);
318     }
319     Point2d* newptr2 = delelement(newptr, delptr-1, n);
320     n--;
321     displaypolygon(newptr2, n);
322     cout << endl;
323
324     delete [] ptr;
325     delete [] newptr;
326
327     return 0;
328 }
```