Practical Exercise 5

Ομάδα : 3

Συμμετέχοντες : Μιχάλης Μιχαήλ
Σώτος Βασιλείου
Πασιουρτίδης Κώστας

Προγραμμα 1

Αρχικα στη main εχω το ανοιγμα του αρχειου και το διαβασμα του. Παιρνω από το αρχειο το N οπου είναι οι γραμμες του αρχειου και στη συνεχεια δεσμευω 2 πινακες μεγεθους N. Οπου ο ενας είναι ο X για τα σημεια X και ο Y για τα σημεια του Y.
Τραβω από το αρχειο τις τιμες για του X και Y και τα αποθηκευω στους εξης πινακες.
Αυξανω τον μετρητη για να πηγαινει σε επομενη θεση τουν πινακα.
Κλεινω το αρχειο.
Και μετα ξεκινα η συναρτηση closestpoints(X,Y,N) οπου στελλω σαν παραμετρους τον πινακα X Y και το μεγεθος τους N.

<u>Ψευδοκωδικας της συναρτησης:</u>

```
ClosestPoints(P[0...n-1])

praksi_min= infinity;

for i=0 to n-2
for j=i+1 to n-1

praksi= sqrt((xi-xj)2+(yi-yj)2);

if praksi < praksi_min
    praksi_min= praksi;

p1=i, p2=j;

cout<<X[p1]<<" "<<Y[p1];
cout << endl;
cout<<X[p2]<<" "<<Y[p2];
```

praksi_min = είναι για να βρισκω τη μικροτερη τιμη αποθηκευω την θεση ώστε να με βοηθησει μετα να τυπωσω τα μικροτερα σημεια.
Αποθηκευω την θεση στα p1, p2 .
Και στα τυπωματα μου αφου τελιωσουν τα for μου, εχω ηδη βρει τα μικροτερα σημεια. Τα τυπωνω με το p1 και p2 αντιστοιχα στους πινακες X και Y.

Προγραμμα 2

```
ifstream fin("ECE325_points.txt")

int N
fin >> N
int i=0, X[N],Y[N]

while(!fin.eof())
	fin >> X[i] >> Y[i]
	i++

fin.close()
Point points[i]

for (int j=0; j<i; j++)
	points[j].x=X[j]
	points[j].y=Y[j]
	convexHull(points, i)
```

Αρχικα στη main ανοιγω και διαβαζω το αρχειο που μας δοθηκε. Παιρνω το N που είναι οι γραμμες του αρχειου και δηλωνω τους δυο πινακες. Οπου ο ενας είναι ο X για τα σημεια X και ο Y για τα σημεια του Y.
Τραβω από το αρχειο τις τιμες για του X και Y και τα αποθηκευω στους εξης πινακες.
Αυξανω τον μετρητη για να πηγαινει σε επομενη θεση τουν πινακα.
Κλεινω το αρχειο.
Δηλωνω τον πινακα points που είναι structure με σημεια X και Y μεσα του.
Αποθηκευω τις τιμες από τους πινακες X και Y στις αντιστοιχες τιμες του structure.

Και μετα ξεκινα η συναρτηση οπου κανει οληη την δουλεια.
Ελεγχω αν υπαρχουν τουλαχιστον 3 σημεια για να λειτουργησει η συναρτηση.
Στην περιπτωση μας υπαρχουν παρα πολλα αρα όλα οκ. Βαζω vector του structure για να αποθηκευω τους X. Στη συνεχεια εχω forloop για να βρω τα μικροτερα σημεια.. Αποθηκευω με push_back στο vector τις τιμες που βρισκω.
If any point 'i' is more counterclock-wise than q, then update q με την μαθηματικη πραξη.
Μετα μπαινει ξανα σε forloop για ελεγχο αν το i is more counterclockwise than current q, then update q με την εξης συναρτηση.
Η εξης συναρτηση κανει μαθηματικες πραξεις για να βρει το val οπου αν το val είναι ==0 είναι γραμμικο και δινει πισω 0. Αν είναι 1 είναι δεξιοστροφο και αν είναι 2 είναι αριστεροστροφο.
Μετα ακολουθουν τα τυπωματα ώστε να εμφανισω το convexhull.

```
void convexHull(Point points[], int n)
{

	if (n < 3) return;
	vector<Point> hull;
	int l = 0;
	for (int i = 1; i < n; i++)
		if (points[i].x < points[l].x)
			l = i;

	int p = l, q;
	do
	{
		hull.push_back(points[p]);

		// Search for a point 'q' such that orientation(p, q,
		// x) is counterclockwise for all points 'x'. The idea
		// is to keep track of last visited most counterclock-
		// wise point in q. If any point 'i' is more counterclock-
		// wise than q, then update q.
		q = (p+1)%n;
		for (int i = 0; i < n; i++)
		{
		// If i is more counterclockwise than current q, then
		// update q
		if (orientation(points[p], points[i], points[q]) == 2)
			q = i
		}

		// Now q is the most counterclockwise with respect to p
		// Set p as q for next iteration, so that q is added to
		// result 'hull'
		p = q

	} while (p != l) // While we don't come to first point

	// Print Result
	for (int i = 0; i < hull.size(); i++)
		cout << "(" << hull[i].x << ", "
			<< hull[i].y << ")\n"  }
```