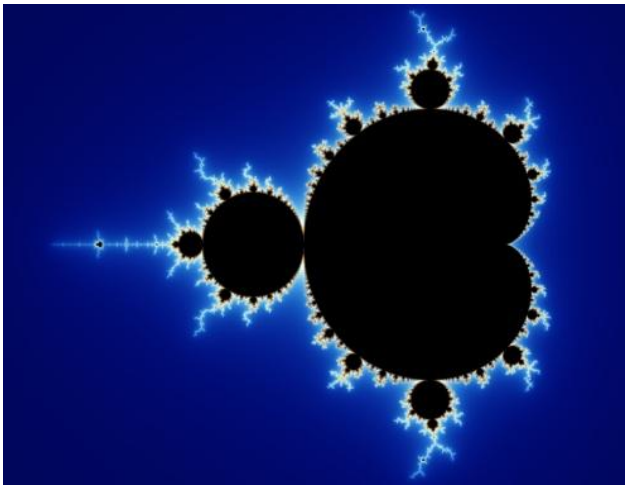
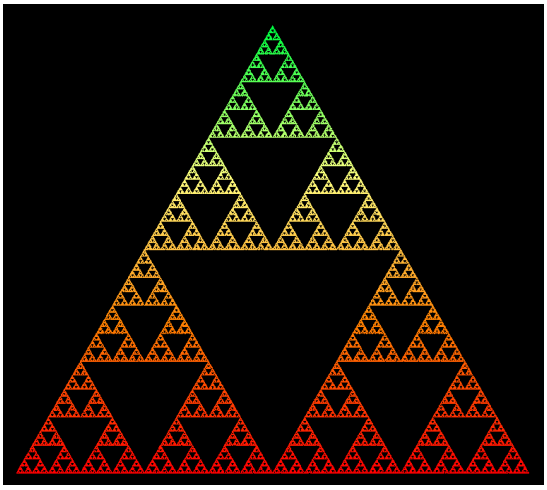
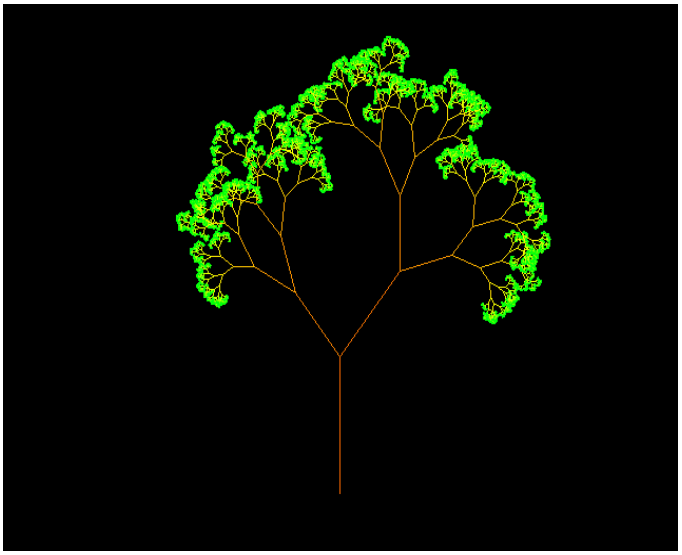
Nombres Complexos i aplicacions als fractals







Mario Rico Fernández

Tutor: Pere Matas

Departament: Matemàtiques

Centre: INS Sa Palomera

Data lliurament: 16-12-2014

Agraïments

Vull agrair al meu tutor de recerca, Pere Matas, tota la seva dedicació. Ell m'ha guiat i m'ha donat idees per poder realitzar aquest treball.

També agraeixo al meu pare, Mario Rico García, que m'ha ajudat sobretot en la part de creació dels programes informàtics.

Per últim, agrair a la meva professora de matemàtiques, Eva Casadevall, que em va descobrir el món dels nombres complexos en les seves classes de batxillerat. I per extensió a tots els professors de matemàtiques que he tingut anteriorment.

Índex

PRESENTACIÓ DEL TREBALL

El món dels nombres complexos em va resultar molt interessant quan els vaig conèixer per primer cop durant el curs de 1r de batxillerat. Consultant alguns llibres em resultava fascinant que una cosa que no existia (el nombre “i” no existeix, és imaginari) pogués tenir tantes aplicacions i ser tan útil.

Vaig pensar que podria ser un bon tema per plantejar el meu treball de recerca. Després de pensar quines aplicacions estaven a l'abast dels meus coneixements, em vaig decidir a estudiar els fractals.

Al mateix temps volia explotar el meu interès per la informàtica i aplicar-lo al treball de recerca. Al principi pensava fer un programa que treballés amb els nombres complexos, però resultava de poca utilitat ja que hi ha molts programes d'ordinador i també calculadores programables que ja feien això que jo volia fer. Així que em vaig decidir a fer un programa que dibuixés el fractal de Mandelbrot, a més de que el meu tutor també ho volia. Per sort, amb el tutor, vam trobar una pàgina web, on tot estava molt ben documentat.

Després, per completar el treball, es tractava de buscar una altre fractal per estudiar i fer el programa informàtic. I ens vam fixar en els arbres. Vaig decidir fer un programa que dibuixés arbres. Al principi eren més simples i simètrics. Després vam millorar els procediments i programes per dibuixar arbres més reals.

Finalment, el tutor va suggerir estudiar com es podia infectar l'arbre. Donat que l'estructura de l'arbre s'assembla molt a les ramificacions dintre d'un pulmó, aquesta simulació es pot aplicar a la infecció causada per una malaltia.

1. NOMBRES COMPLEXOS

Els nombres complexos són una extensió dels nombres reals, en que hi ha una unitat imaginària i una altre sense, podríem dir que es la part normal.

$$\mathbb{C}\ \text{Complexos}\begin{cases}\mathbb{R}\quad \text{Reals}\begin{cases}\mathbb{Q}\quad \text{Racionals}\begin{cases}\mathbb{Z}\quad \text{Enters}\begin{cases}\mathbb{N}\quad \text{Naturals}\\ \text{Enters negatius}\end{cases}\\ \text{Fraccionaris}\end{cases}\\ \text{Irracionals}\end{cases}\\ \text{Imaginaris}\end{cases}$$

En els nombres Reals, existeixen algunes operacions que no es poden resoldre i que amb els Complexos si que es poden solucionar, per exemple les arrels quadrades o d'índex parell negatives, per tant els nombres Complexos s'utilitza la següent relació: $i^2 = -1$

Estan compostos per una part imaginaria, que és característica d'aquest conjunt de nombres i la part real, que és la que no té part imaginaria si no que només esta compost per un nombre real.

Per que es van inventar?

Es van inventar per calcular operacions que eren impossibles de calcular amb els nombres reals, com per exemple les arrels quadrades negatives, descomposició de polinomis, resoldre equacions algebraiques...

1.1 Operacions amb nombres Complexos

Algunes de les operacions amb els nombres Complexos es poden resoldre tant amb una notació cartesiana, com una notació polar. Això varia depenent de l'operació que es realitzi.

Hi destaquem les següents:

Suma i resta

Per sumar dos nombres complexos s'ha d'utilitzar la notació cartesiana.

Es sumen les components reals dels sumands i les components imaginàries per separat:

$$a + bi + a' + b'i = (a + a') + (b + b')i$$

Per restar es fa de la manera següent:

$$a + bi - (a' + b'i) = a + bi + (-a') + (-b')i = (a - a') + (b - b')i$$

Multiplicació

Per multiplicar dos nombres complexos es pot utilitzar qualsevol de les dues notacions:

- Notació cartesiana:

$$(a + bi) \cdot (a' + b'i) = a \cdot a' + a \cdot b'i + bi \cdot a' + bi \cdot b'i$$

- Notació polar

$$r_{\phi} \cdot r'_{\phi'} = r \cdot r'_{\phi + \phi'}$$

Divisió

Per dividir dos nombres complexos s'utilitza normalment la notació polar, per ser la forma més fàcil. Tot i així també es pot operar amb la notació cartesiana.

- Notació cartesiana

$$\frac{a + b\mathbf{i}}{a' + b'\mathbf{i}} = \frac{(a + b\mathbf{i}) \cdot (a' - b'\mathbf{i})}{(a' + b'\mathbf{i}) \cdot (a' - b'\mathbf{i})} = \frac{a \cdot a' - a \cdot b'\mathbf{i} + b\mathbf{i} \cdot a' - b\mathbf{i} \cdot b'\mathbf{i}}{a'^2 - (b'\mathbf{i})^2} =$$

$$= \frac{(a \cdot a' + b \cdot b') + (b \cdot a' - a \cdot b')\mathbf{i}}{a'^2 + b'^2}$$

- Notació polar

$$\frac{r_{\phi}}{r_{\phi'}} = \left(\frac{r}{r'}\right)_{\phi - \phi'}$$

Potència

Per fer el quadrat d'un nombre complex, cal realitzar el següent:

- En notació cartesiana, cal utilitzar el Binomi de Newton; concretament, el quadrat (en potència de 2) és:

Aquest procediment és llarg de fer, si a més l'utilitzem en potencies superiors a 2 o 3.

$$(a+b\mathbf{i})^2 = (a+b\mathbf{i})\cdot(a+b\mathbf{i}) = (a\cdot a - b\cdot b) + (a\cdot b + b\cdot a)\mathbf{i} = (a^2-b^2) + (2ab)\mathbf{i}$$

Per això s'utilitza majoritàriament la forma o notació polar.

- En notació polar i generalitzant

$$(r_\phi)^n = r^n_{\phi\cdot n}$$

n = Exponent

Arrels

El nombre d'arrels *n-èsimes* d'un nombre complex és exactament *n*.

L'arrel n-èsima d'un nombre complex en forma polar $\sqrt[n]{r}_\propto$ és un altre nombre complex que en forma polar verifica que:

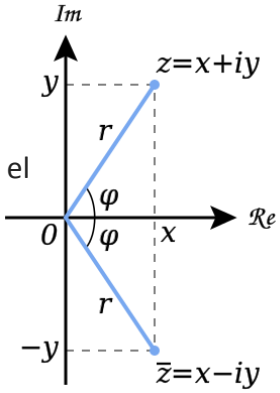
- El seu mòdul és l'arrel n-èsima del mòdul del nombre donat.

- El seu argument és $\beta = \frac{\propto + k\cdot 360^\circ}{n}$, amb k = 0,1,2... n-1

$$\sqrt[n]{r}_\propto = (\sqrt[n]{r})_{\frac{\propto + k\cdot 360^\circ}{n}}$$

Conjugat

El conjugat d'un nombre complex té el mateix mòdul i angle oposat.



Les parts reals i imaginàries d'un nombre complex poden ser extretes usant el conjugat:

$$\operatorname{Im}(z) = \tfrac{1}{2i}(z-\bar{z}).\operatorname{Re}(z) = \tfrac{1}{2}(z+\bar{z}),$$

A més, un nombre complex és real si només el seu conjugat és igual a ell.

1.2 Teorema Fonamental de l'Àlgebra

Diu que tots els polinomis de coeficients complexos i de grau n (n≥1) tenen n arrels.

Tot polinomi en una variable tot i ser no constant i amb nombres complexos, té tantes arrels com indica el grau.

Aquest resultat és fonamental perquè demostra que el cos dels nombres complexos és un cos *algebraicament tancat*, a diferència del cos dels nombres reals.

Com que el quadrat d'un nombre real no és mai negatiu, llavors hi ha equacions de segon grau com, per exemple, $x^2 = -1$, que no tenen solució entre els nombres reals. En introduir els nombres complexes si que trobem solucions per aquestes equacions. Les dues solucions de l'equació anterior serien x = i, i x = -i

Però, encara més, en treballar al conjunt dels nombres complexos, resulta que tenim solucions per totes les equacions siguin del grau que siguin.

Una equació de grau n té n solucions en C (comptades amb la seva multiplicitat)

Per tant, un polinomi de grau n sempre es pot descompondre en factors lineals (factors de grau (1)

1.3 Funcions Complexes

Hi ha una relació entre les funcions trigonomètriques i les funcions exponencials i logarítmica.

La funció exponencial e^xte la seva definició dintre dels nombres complexos

Si z = x + yi , llavors $e^z = e^x(\cos y + i \sin y)$

Això és clarament una extensió de la definició real ja que si y=0, dóna e^x

Aquesta definició permet conservar totes les propietats importants de la funció exponencial , i també és interessant perquè relaciona funcions matemàtiques tan importants com la exponencial, el sinus i el cosinus.

Per exemple, calcularem $e^{1+\pi i}$, $e^{\frac{\pi}{2}i}$

a) $e^{1+\pi i} = e^1(\cos\pi + i \sin\pi) = -e$

b) $e^{0+\frac{\pi}{2}i} = e^0\left(\cos\frac{\pi}{2} + i \sin\frac{\pi}{2}\right) = i$

Si calculem $e^{\pi i}$, trobarem la relació que hi ha entre els cinc números més importants de les matemàtiques 0, 1, π, *e, i*

$$1 + e^{\pi i} = 0$$

Propietats de la funció exponencial de variable complexa

Exposem ara, sense demostració, algunes propietats de la funció e^z

I e^z mai pot ser zero

II Si x és real, llavors $\left|e^{ix}\right| = 1$

III $e^z = 1$ si i només si z és un múltiple enter de $2\pi i$

IV La derivada (complexa) de e^z és e^z

2. FRACTALS

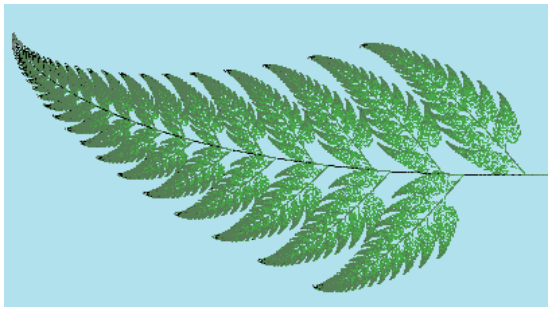
Un fractal és un objecte matemàtic molt complex compost per algorismes. L'objecte al augmentar-lo es va repetint el mateix, tant la mida com la forma.

Els fractals neixen de l' intent de trobar una geometria més apropiada per descriure els objectes de la natura, aquestes formes complexes van ser estudiades per Benoit Mandelbrot.

La paraula "fractal" prové del terme fraccionari, ja que d'una porció petita podem tornar a trobar la mateixa figura i així successivament.

2.1 Fractals a la vida quotidiana

- Estructures a la vida quotidiana que segueixen el fractal:



http://blocs.xtec.cat/eliesmates/?tag=fr 1

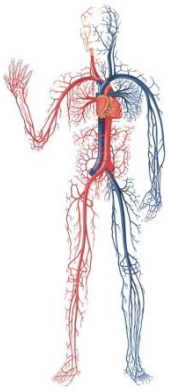
Estructura d'algunes fulles

http://planta0.wordpress.com/2013/12/17/ 1

Forma de les flors

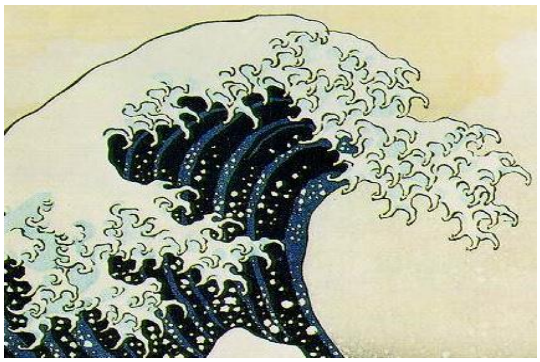
http://cat.bloctum.com/mesogio/page/17/ 1

Llamp



http://cienciadesofa.com/2013/08/fractal 1

Ramificació de venes i artèries



http://terraxaman.blogspot.com.es/2009/0 1

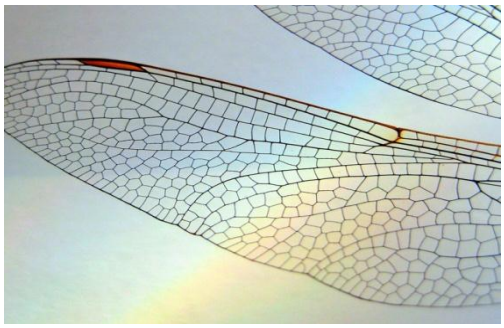
Onades

http://comvisualfractales.blogspot.com.e 1

Costa de les platges

http://www.curiosaweb.com/wp-content/upl 1

Closques de cargols i mol·luscs



http://distraete.com/pense-que-la-natura 1

Ales dels insectes

http://www.ddecoracion.com/guia/espejos/ 1

http://bitacorascubanas.com/toda-histori 1

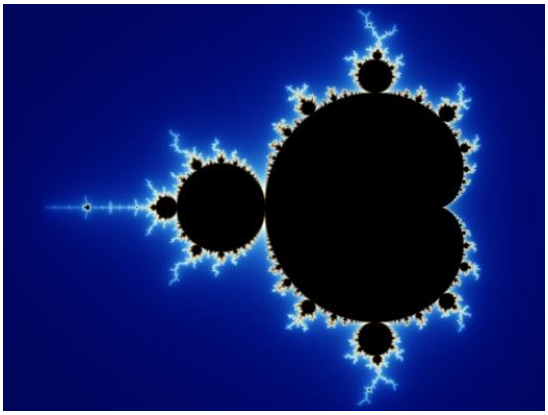
Reflex dels miralls

Totes aquestes imatges tenen en comú una cosa, que totes són fractals, ja que es van repetint consecutivament, agafant una petita part de la imatge original, al augmentar-la obtindriem la mateixa imatge que teniem abans.

Com podem veure en l'exemple dels miralls, en l'estructura d'algunes plantes i fins i tot en les ales dels insectes, es compleix aquesta norma.

2.2 Fractals Populars

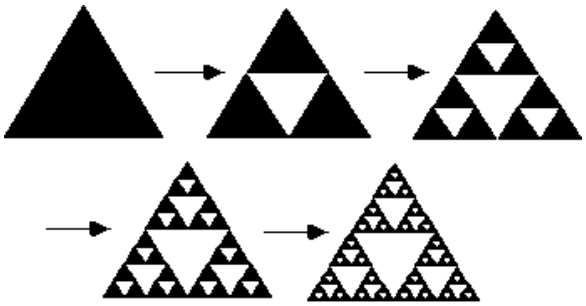
Mandelbrot



- El conjunt de Mandelbrot és un dels fractals més famosos i reconeguts, avui dia és un dels principals objectes d'estudi de la dinàmica complexa.

Fins que no van aparèixer els primers ordinadors digitals no es va poder veure aquest fractal $Z = Z^2 + C$ en tota la seva complexitat.

Triangle de Sierpinski



http://math.bu.edu/DYSYS/chaos-game/node 1

- El triangle de Sierpinski és un dels exemples bàsics de conjunt auto-semblant, una de les propietats fonamentals dels fractals. Encara que va ser construït inicialment a partir d'un triangle equilàter, anomenat "*triangle de Sierpiński*" *canònic*, es pot fer la construcció a partir de qualsevol triangle.

2.3 Mandelbrot

Aquest fractal serà objecte d'estudi dins del treball de recerca, ja que, com he explicat anteriorment, és un dels fractals més importants i coneguts.

Per cada valor c dels nombres complexos considerem la successió $(Zn)_n$ on $Z_0= 0$ i $Z_{n+1} = Z_n^2 + C$, i estudiem si aquesta successió convergeix, o amb quin grau de velocitat divergeix.

Segons aquest resultat, pintarem el punt C d'un color o d'un altre i així podrem fer una replica del dibuix original.

2.3.1 Descripció

Una funció matemàtica és una expressió que permet calcular una variable dependent del valor de les altres, procés anomenat iteració.

Per exemple, anem a iterar tres vegades la funció $Z_{n+1} = Z_n^2 + C$, on el valor inicial de Z és Z_0:

- Primera iteració: $Z_1 = Z_0^2 + C$

- Segona iteració: $Z_2 = Z_1^2 + C$

- Tercera iteració: $Z_3 = Z_2^2 + C$

Llavors, podem obtenir Z_3 que correspon al valor de Z després de fer les tres iteracions i així successivament.

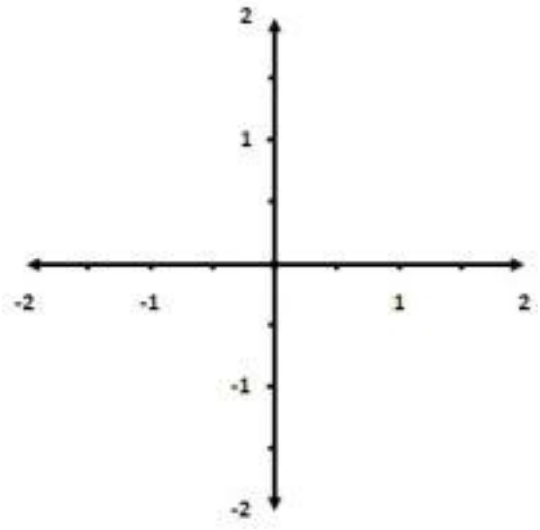
Tot seguit, quan ja tenim enllestit la funció i el nombre d' iteracions que volem fer, ho hem de representar en forma de dibuix.

- Primer pas: Escollir la funció, en aquest cas $Z_{n+1} = Z_n^2 + C$

- Segon pas: Operar amb els nombres complexos, que com ja sabem consten d'una part real i una altra d'imaginària. On "a" és la part real i "b" un nombre real que ve acompanyat de la unitat imaginària "i" i formen la part imaginària.

De tal manera que el nombre complex queda representat en forma bionòmica com Z = (a+bi)

- Tercer pas: Establir un pla de coordenades



- Quart Pas: Fixar totes les condicions de càlcul per obtenir el fractal que volem. Per exemple, el de Mandelbrot:

- El valor inicial de Z serà $Z_0 = (0+0i)$.

- El valor de C correspon a las coordenades d'un punt del pla, C = (X+Yi).

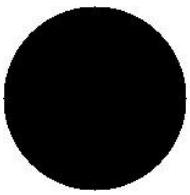
- El nombre d'iteracions pot ser qualsevol valor enter i positiu.

- El Radi ha d'estar fixat a 2, perquè llavors faríem càlculs innecessaris, a més de que nombres més grans ja no pertanyen al conjunt.

Finalment hem de representar-ho, quan ja tenim tots els passos següents en compte.

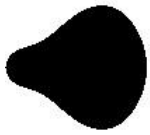
El procés consisteix en analitzar cada punt del pla per esbrinar si pertany al conjunt o si es troba fora d' ell. La regla a aplicar és que si aconseguim iterar la funció el nombre de vegades que hem fixat sense que el valor absolut de Z superi el radi d'escapament, el punt es troba dins del conjunt, en cas contrari considerem que el punt divergeix quedant fora del conjunt.

I per tant, si el nostre punt esta dins del conjunt, és a dir, que convergeix, el pintem d'un color, en aquest cas negre. Però el nombre d'iteracions varia la precisió del conjunt de Mandelbrot:

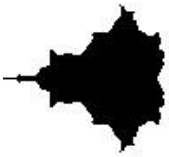


2 iteracions

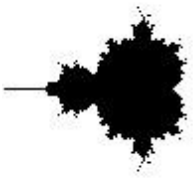
Si pintéssim tots els punts que convergeixen del mateix color i només intervenen 2 iteracions, així seria com quedaria el conjunt de Mandelbrot, però el conjunt es por iterar fins a un màxim de 16 iteracions, per veure-ho molt més definit:



4 iteracions



8 iteracions



16 iteracions

Una cosa curiosa que hem dit abans es que el màxim nombre d'iteracions és 16, si nosaltres en el programa li diguéssim que faci més iteracions, per exemple 20, 30, 40… no canviaria res, el dibuix seria igual.

2.3.2 Explicació del programa

Anirem mirant punt per punt del pla complex, per veure si la successió generada és convergent o divergent.

(Cx,Cy) es un punt pla complex, Cx + Cyi

La successió la calcularem en la variable (Zx,Zy) que representa el nombre Zx + Zyi

Al principi Zx= 0 i Zy= 0 (Z= 0 + 0i) i el $Z_{nou}= Z^2 + C = (Zx + Zyi)^2 + (Cx + Cyi)$

Desenvolupem i queda:

$Z_{nou} = Zx^2 + 2i\ ZxZy + Zy^2\ i^2 + Cx + Cyi$

Com que $i^2 = -1$, agrupem la part real i la part imaginària Z_{nou} 0 $(Zx^2 – Zy^2 + Cx) + (2\ Zx\ Zy + Cy)$

Per tant, tindrem:

$Zx_{nou} = Zx^2 – Zy^2 + Cx$

$Zy_{nou} = 2\ Zx\ Zy + Cy$

I els resultats els tornem a ficar a les variables Zx, Zy per tornar a fer l'operació (bucle).

$Zx = Zx_{nou}$

$Zy = Zy_{nou}$

Ara arriba el moment de comprovar la convergència o divergència de la successió. Si alguna Z se surt de la pantalla, la successió es divergent.

Per això utilitzem una variable "converge" que en principi es certa, però si en algun pas passa que $\sqrt{Zx^2 + Zy^2} > 2$, posem que "converge" es Falsa.

En el programa, la condició l'hem expressat:

$Zx^2 + Zy^2 > 4$

Per no haver de calcular tantes arrels innecessariament, ja que no les necessitem.

Si convergeix es pinta el punt de color negre.

Si divergeix va canviant els colors segons el moment en que s'ha trobat la divergència (Segons la N).

Esquema senzill del programa

- Per cada (Cx, Cy) de la pantalla

  - Inicialitzar Zx, Zy a (0,0)

    - Des de 1 fins N

      (Calcular Zx, Zy següents i veure si divergeix)

      (Si es detecta divergència pintar el punt del color adequat segons el valor que s'ha detectat aquesta divergència)

2.3.3 Codi del programa

Aquesta instrucció la posa per defecte el visual basic. Obliga a declarar totes les variables que s'utilitzen

Option Explicit

N és el nombre d'iteracions en fer la successió de nombres complexos. L'hem posada a 16. És una constant

Const N = 16

Declarem les variables que defineixen l'amplada i l'alçada de la pantalla. No s'han posat com a constants ja que si volem millorar el programa per que faci un zoom, aquests valors canviarien.

Dim XMIN As Double

Dim XMAX As Double

Dim YMIN As Double

Dim YMAX As Double

El programa se inicia amb el procediment Form_Activate. Form es refereix a la finestra principal del programa (el visual basic anomena formularis a les finestres). Quan s'activa (amb el mètode Activate) crida al subprograma "mandelbrot"

Private Sub Form_Activate()

mandelbrot

End Sub

D'aquí i fins al final tenim el programa que fa el dibuix. Es diu mandelbrot

Private Sub mandelbrot()

Dim cx As Double, cy As Double	(Cx, Cy) es el punt inicial de la successió.
Dim zx As Double, zy As Double	(Zx, Zy) són els punts successius
Dim zxnou As Double	Zxnou es una variable auxiliar que fem servir
Dim i As Integer	per calcular el següent (Zx, Zy)
Dim converge As Boolean	i ens indica la iteració que estem fent
	converge es booleana, pot ser si o no.
	Ens indica si la successió convergeix
XMIN = -2	Com que utilitzem la pantalla del mini portàtil
XMAX = 2	Amplada/Alçada = 1024 : 600
YMIN = -1.125	Si fixem l'amplada de x= -2 a x= 2
YMAX = 1.125	queden aquestes Y

Amb la funció Scale es defineixen les dimensions de la pantalla indicant la cantonada superior esquerra i la cantonada inferior dreta. Això serveix per dibuixar punts amb PSet(X,Y)

```
Scale (XMIN, YMAX)-(XMAX, YMIN)
cx = XMIN
Do While cx < XMAX                    Comencem els càlculs !
  cy = YMIN                           Amb aquest doble bucle fent variar cx i cy en tots els
  Do While cy < YMAX                  valors de x i de y. Cada bucle acaba en els seu Loop
    zx = 0
    zy = 0                            La successió zn comença en (0,0) = 0 + 0i complex
    i = 0
    converge = True
    Do While i <= N And converge      Aquest és el bucle per fer les N= 16 iteracions
      zxnou = zx ^ 2 - zy ^ 2 + cx
      zy = 2 * zx * zy + cy           Calcula el zn següent  = zn^2 + c
      zx = zxnou                      el zn està desglossat en la part real zx i la imaginaria zy

      If (zx) ^ 2 + (zy) ^ 2 > 4 Then  Si el mòdul de zn és major que 2 (se surt de la
                                       pantalla, entenem que divergeix

        If i Mod 4 = 0 Then PSet (cx, cy), vbBlack    i indica a quina iteració se surt de la
        If i Mod 4 = 1 Then PSet (cx, cy), vbBlue          pantalla.
        If i Mod 4 = 2 Then PSet (cx, cy), vbGreen    Segons la i, pintem d'un color diferent
        If i Mod 4 = 3 Then PSet (cx, cy), vbRed      Mod és el residu de la divisió entera
        converge = False
      End If
     i = i + 1
    Loop
    If converge Then PSet (cx, cy), vbBlack   Si després de les iteracions hi ha convergència
                                              pintem de negre el punt
```

cy = cy + 0.003

Loop

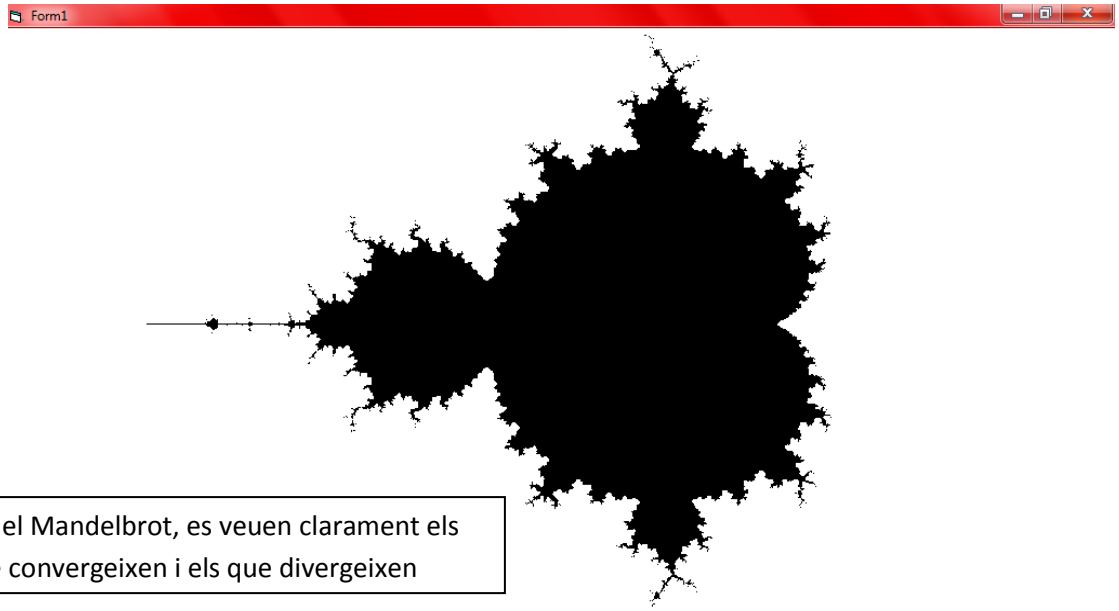
cx = cx + 0.003 *Aquest dos Loop's tanquen el doble bucle inicial*

Loop *L'increment és 0,003 després de fer proves amb varis valors i valorar el resultat del programa*

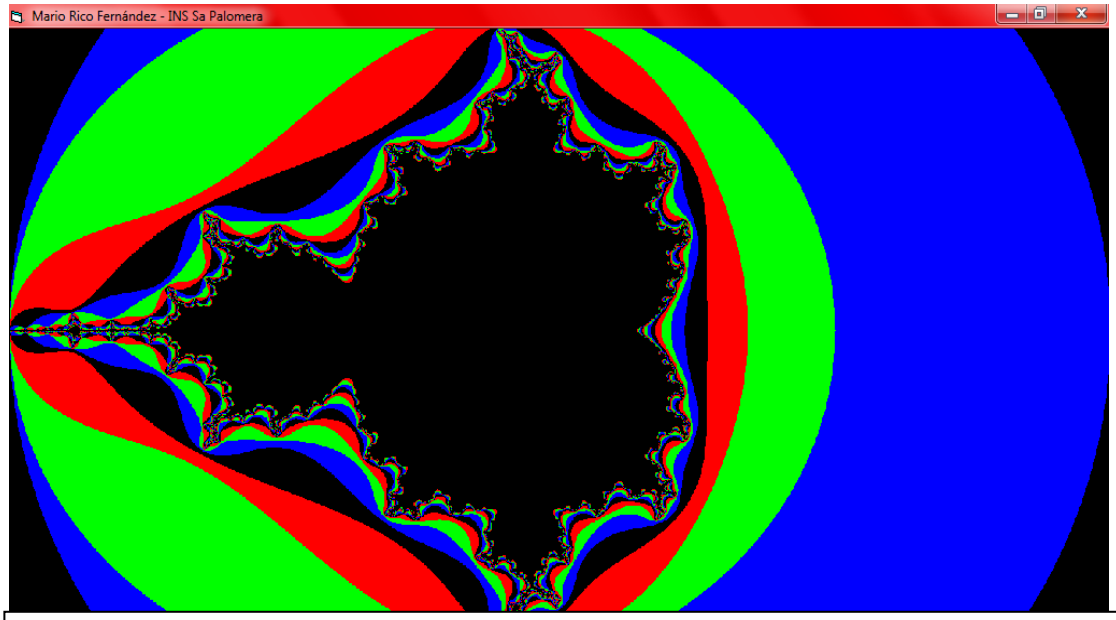
(qualitat del dibuix < - > velocitat d'execució)

End Sub

2.3.4 Resultats



Resultat del Mandelbrot, es veuen clarament els punts que convergeixen i els que divergeixen



Es veuen les diferents velocitats, quan els punts divergeixen, només està fet amb quatre colors i per tant es van repetint successivament els quatre colors (negre, vermell, verd i blau)

2.3.5 Dificultats

Al començament el programa estava escrit amb llenguatge Scratch, però vaig tenir molts problemes ja que era un llenguatge que no tenia massa potencia de càlcul i al obrir el programa de Mandelbrot trigava unes hores en executar-se completament i això complicava molt la tasca.

Per això vaig decidir fer servir el llenguatge Visual Basic que és més complet alhora de calcular i més eficaç alhora de fer repeticions i bucles.

3. ARBRES

A la natura, un exemple clàssic de fractals el trobem als arbres. Un arbre es composa d'un tronc del qual surten branques, i d'aquesta surten sub branques més primes i així successivament.

3.1 Introducció

Ens plantegem fer un programa que dibuixí arbres. La manera de estructurar-lo serà per nivells. El nivell 0 és el tronc. El nivell 1 són les branques que surten del tronc, les branques primàries. El nivell 2 són les sub branques que surten de les branques primàries,etc.

Hem d'observar que si ens situem a un determinat nivell, per exemple, el nivell 2 en una determinada sub branca el treball que hem de fer es equivalent a dibuixar un arbre sencer, ja que té la mateixa estructura.

També hem de pensar que, a mesura que avancem de nivell, les branques i sub branques es van reduint de longitud per donar a la construcció un aspecte d'arbre.

Més endavant, quan l'arbre ja estigui construït, ens plantejarem fer l'estudi d'una infecció que pugui afectar a l'arbre. La infecció començarà al tronc i s'anirà estenent per algunes branques.

La infecció dependrà dins determinats paràmetres, que anirem variant per observar els resultats i fer-ne un estudi estadístic.

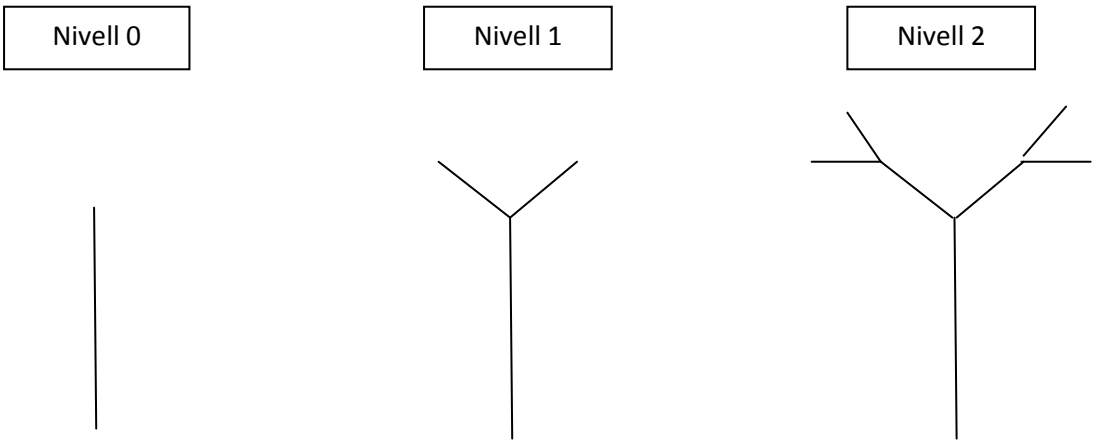
3.2 Programa que genera arbres simètrics amb dues branques

El primer problema que ens plantegem és dibuixar un arbre que a cada nivell generem dues sub branques. És a dir, del tronc surten dues branques, de cada branca sortiran dues sub branques i així successivament.

L'angle que forma unint aquestes branques també serà fixat: branca i dues sub branques formaran 120º= 2π/3 rad, entre si. Expressem els graus en radiants perquè el programa treballa en radiants.

A més, per donar aspecte jeràrquic, les sub branques tindran una longitud més petita que la branca mare o principal.

Aquest efecte es realitzarà dintre del programa mitjançant un factor de reducció:



3.2.1 Plantejament

La principal acció que hem de fer al programa és:

A partir d'una branca, construir les dues branques filles. Aquesta acció, repetida d'una manera organitzada, ens permetrà dibuixar l'arbre sencer.

Estudiarem la relació que té la branca mare amb la primera branca filla. Cal fixar-nos que, per la forma del problema, ens va molt bé treballar amb coordenades polars en lloc de coordenades cartesianes. Sabem la longitud de les branques i els angles que formen.

	Branca Mare	Primera branca filla	Segona branca filla
Longitud	L	L · P	L · P
Angle	α	α +180 + 120º	α +180 + 240º

Si la branca mare té les següents coordenades polar L_α on L és la longitud i α l'angle que forma amb el semi eix horitzontal positiu.

Farem ús de la següent propietat dels nombres complexos.

Multiplicar per un número complex r_o té el següent significat geomètric:

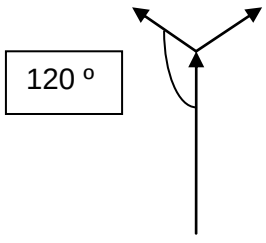
- El mòdul del nombre original queda multiplicat per r
- A l'angle del nombre original se li suma o

Per aplicar aquesta propietat hem de considerar l'oposat de la branca mare, per que tots tres vectors tinguin el mateix origen.

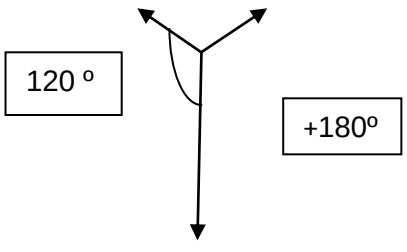
Com que volem treballar amb nombres complexos, el que hem de fer és pensar com que cada branca és un vector i tots tres vectors (la branca mare i les dues branques filles) han de sortir del mateix punt inicial.

Però, hi ha un petit problema, en realitat el que nosaltres volem és que l'arbre es comenci a dibuixar des del punt 0 de la branca principal i no al punt 1 d'aquesta.

Bé, això es pot solucionar sumant-li al vector de la branca principal 180º, perquè el vector es giri com nosaltres volem.



Tots tres vectors parteixen del mateix punt



Donem la volta al vector principal

Un altre característica que hem d'aplicar i que ens és molt útil és aplicar les propietats dels nombres complexos com escriure un nombre complex en forma polar, com està explicat en el requadre de més amunt.

Sabem que al multiplicar dos nombres en forma polar queda el producte d'aquests dos factors i l'angle que dona és el resultat dels dos angles dels dos factors sumats.

$$L_{\alpha} \cdot r_{\theta} = (L \cdot r)_{\alpha+\theta}$$

En el programa hem posat que r⟶ P i també sabem que θ⟶ 120º, perquè cada vegada surtin branques més curtes en cada nivell que augmentem hem de posar que r<1.

3.2.2 Programa

Com hem explicat molt breument en l'apartat anterior, per canviar el sentit de la branca principal sumem 180ª a l'angle i considerem $L_{\alpha+180} = L_{\pi}$ llavors la primera sub branca és $L \cdot P_{\alpha+\pi+2\pi/3}$

I de la mateixa manera, la segona sub branca és $L \cdot P_{\alpha+\pi+4\pi/3}$

D'aquesta manera de cada branca sabem la seva longitud i el seu angle. Però per dibuixar-la també hem de saber el seu punt inicial.

A partir d'aquesta idea volem plantejar un procediment o sub rutina que dibuixi el sub arbre corresponen a una determinada branca.

També hem de plantejar acabar el arbre en un determinat nivell.

Si tenim ja dibuixat una branca amb les següents dades:

- Punt Inicial (P)

- Longitud (L)

- Angle (α)

- Punt Final (Q)

Amb aquesta funció: sub arbre (Ax,Ay,L,Alpha,nivell)

Podem dibuixar el propi arbre, ja que la primera branca és el tronc amb nivell 1.

El tronc, dibuixant-lo amb els comandos del Visual Basic és ***Line*** (0,0) – (0,1)

I la instrucció seria:

Sub arbre (0,1,1,π/2, 1)

(0,1) ⟶ Punt Final

(1) ⟶ Longitud

(π/2) ⟶ Angle

(1) ⟶ Nivell

3.2.3 Funció recursiva

Escrivim el procediment del sub arbre (Ax as Double, Ay as Double, L as Double, Alpha as Double, nivell)

Ax as Double, Ay as Double són el punt final de la branca.

Fixeu-vos que he de controlar en quin nivell estem, sinó el programa no acabaria mai.

N ⟶ és una constant amb el nivell màxim (exemple 20)

P ⟶ és el factor de reducció de longitud (exemple 0.5)

Si el nivell = N ja no és creen més sub branques.

La primera sub branca tindrà la següents característiques:

- Punt Inicial ⟶ (Ax,Ay)

- Longitud ⟶ L·P

- Angle ⟶ Alpha + π + 2π/3

Calculem el punt final

$P_{final} = P_{inicia} + (Vx,Vy) = P_{inicial} + L \cdot P$ (cos (Angle), sin (Angle))

I dibuixem amb la comanda ***Line***

Però aprofitem que el procediment està fet de manera general i el cridem aquí mateix, però augmentant en 1 el nivell.

Amb la instrucció del Visual Basic: ***Call*** sub arbre

La segona sub branca

- Punt Inicial ⟶(Ax,Ay)

- Longitud ⟶L·P

- Angle ⟶Alpha + π + 4π/3

De la mateixa manera calculem el punt final

Vx = L·P (cos (Angle)

Vy = L·P (sin (Angle)

Punt Final = (Ax,Ay) + (Vx,Vy)

Per dibuixar la branca: Line (Ax,Ay) – (Ax + Vx, Ay + Vy)

I tornem a cridar a programa sub arbre

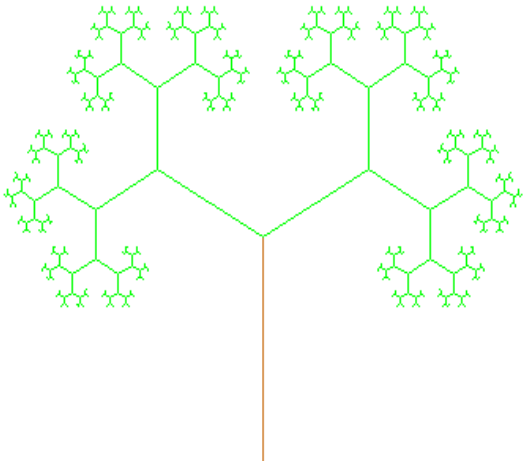
3.2.4 Codi del programa

```
Option Explicit
Const N = 10
Const P = 0.5
Const PI = 3.141592
Private Sub dibuixasubarbre(Ax As Double, Ay As Double, L As Double, Alpha As
Double, nivell As Integer)
Dim Vx As Double
Dim Vy As Double
If nivell = N Then Exit Sub
Vx = L * P * Cos(Alpha + PI + 2 * PI / 3)
Vy = L * P * Sin(Alpha + PI + 2 * PI / 3)
If nivell > 4 Then
   Line (Ax, Ay)-(Ax + Vx, Ay + Vy), vbGreen
Else
  Line (Ax, Ay)-(Ax + Vx, Ay + Vy), RGB(204, 119, 34)
End If
Call dibuixasubarbre(Ax + Vx, Ay + Vy, L * P, Alpha + PI + 2 * PI / 3, nivell + 1)
Vx = L * P * Cos(Alpha + PI + 4 * PI / 3)
Vy = L * P * Sin(Alpha + PI + 4 * PI / 3)
If nivell > 4 Then
   Line (Ax, Ay)-(Ax + Vx, Ay + Vy), vbGreen
Else
```

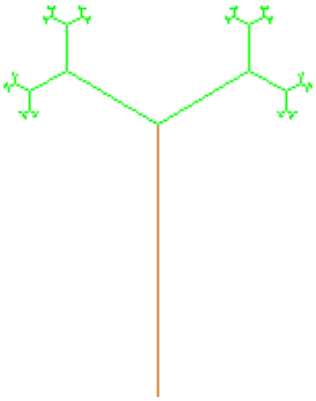
```
  Line (Ax, Ay)-(Ax + Vx, Ay + Vy), RGB(204, 119, 34)
End If

Call dibuixasubarbre(Ax + Vx, Ay + Vy, L * P, Alpha + PI + 4 * PI / 3, nivell + 1)
End Sub

Private Sub Form_Activate()
Dim Px As Double
Dim Qx As Double
Dim Py As Double
Dim Qy As Double
Dim Angle As Double
Dim Longitud As Double

Scale (-4, 4)-(4, 0)

Px = 0
Py = 0
Longitud = 1
Angle = PI / 2
Qx = Px + Longitud * Cos(Angle)
Qy = Py + Longitud * Sin(Angle)
Line (Px, Qx)-(Py, Qy), RGB(204, 119, 34)
Call dibuixasubarbre(Py, Qy, Longitud, Angle, 1)
End Sub
```

Resultats

Un cop explicat tot el programa, ara mirarem els resultats obtinguts variant las constants P (la relació de mida entre branca mare i la filla) i N (nombre de nivells en que sortiran branques):

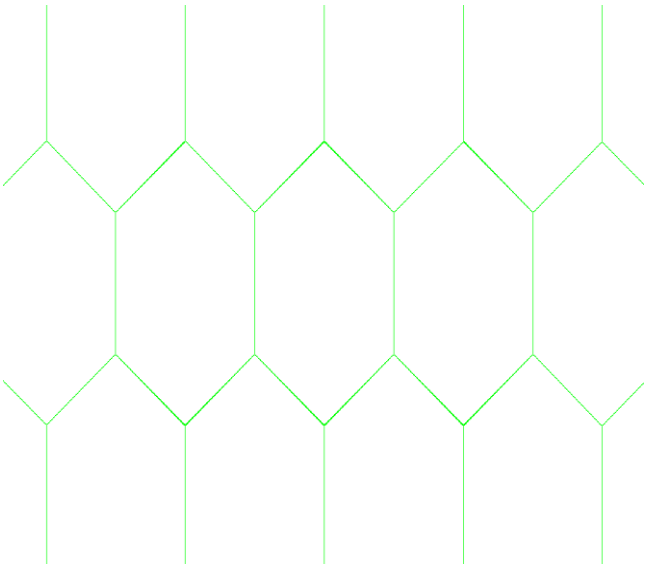


N= 10 P= 0.6

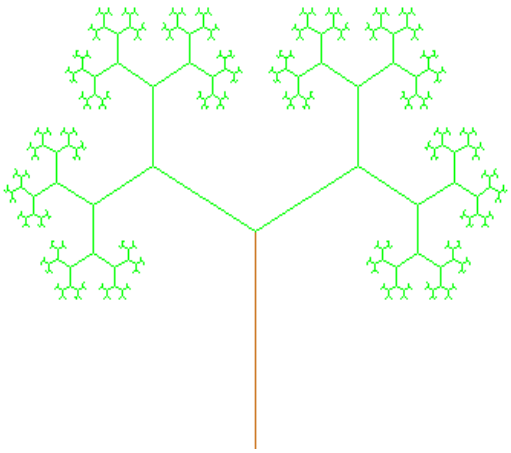


N= 10 P= 0.4

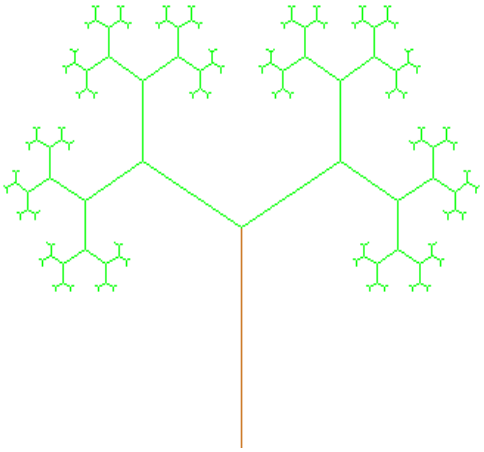
N= 10 P= 0.1



N= 10 P= 1

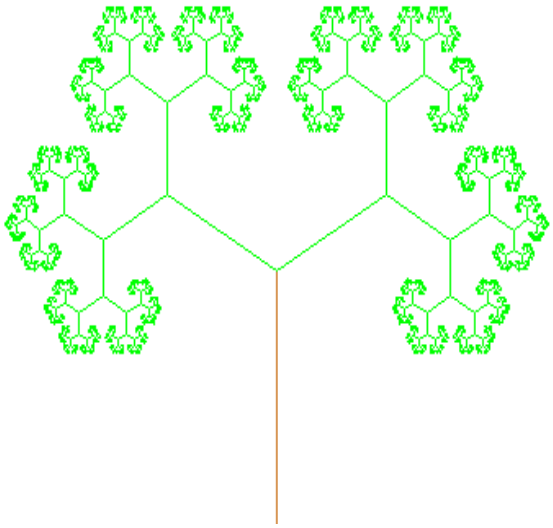


N = 10 P = 0.6



N = 8 P = 0.6

N = 1 P = 0.6



N = 15 P = 0.6

3.3 Programa que genera branques a l'atzar

3.3.1 Plantejament. Angle aleatori

Ara volem modificar el programa anterior per que l'arbre generat no tingui un aspecte tant simètric i artificial. Ho farem canviant els angles fixes que formen les branques (120º i 240º) per altres angles que generi el programa aleatòriament.

3.3.2 Programa

A la funció recursiva sub arbre crearem una nova variable anomenada AngleAleat que substituirà els llocs on hi ha 2 pi / 3 rad (=120º) i 4 pi / 3 (= 240º)

Farem servir la instrucció de visual basic Rnd(). Aquesta funció genera un nombre aleatori entre 0 i 1.

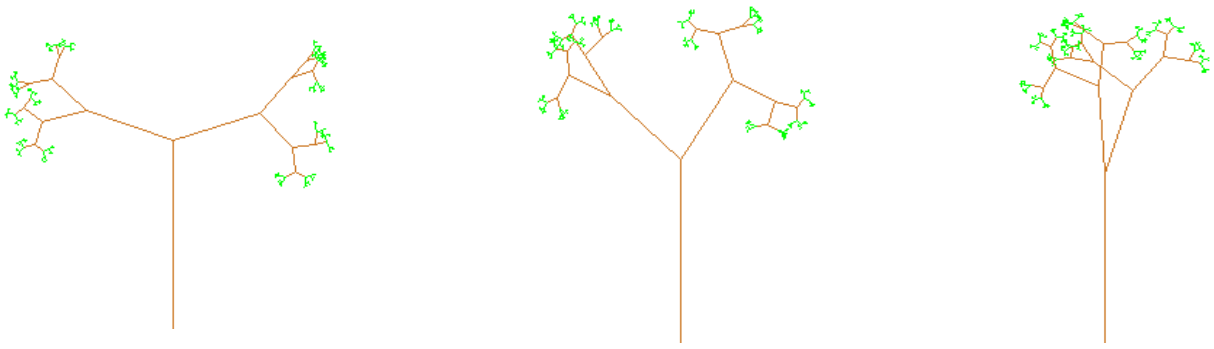
Com que volem substituir 2 pi /3 per un angle entre pi/2 i pi , la instrucció que posarem serà

anglealeat = Alpha + PI + PI / 2 + Rnd() * (PI - PI / 2)

Com que volem substituir 2 pi /3 per un angle entre pi i 3 pi /2 , la instrucció que posarem serà

anglealeat = Alpha + PI + PI + Rnd() * (3 * PI / 2 - PI)

Amb aquest canvi generem arbres d'aquesta forma:



Volem millorar l'aspecte de l'arbre per que en lloc de que cada branca tingui 2 sub branques, en tingui quatre, amb aquestes característiques.

Tot això es pot compactar en el següent bucle:

```
For densitat = 0 To 3
anglealeat = Alpha + PI + PI / 2 + densitat * PI / 4 + Rnd() * (PI / 4)
Vx = L * P * Cos(anglealeat)
Vy = L * P * Sin(anglealeat)
Line (Ax, Ay)-(Ax + Vx, Ay + Vy), vbGreen
Call dibuixasubarbre(Ax + Vx, Ay + Vy, L * P, anglealeat, nivell + 1, 0)
Next
```

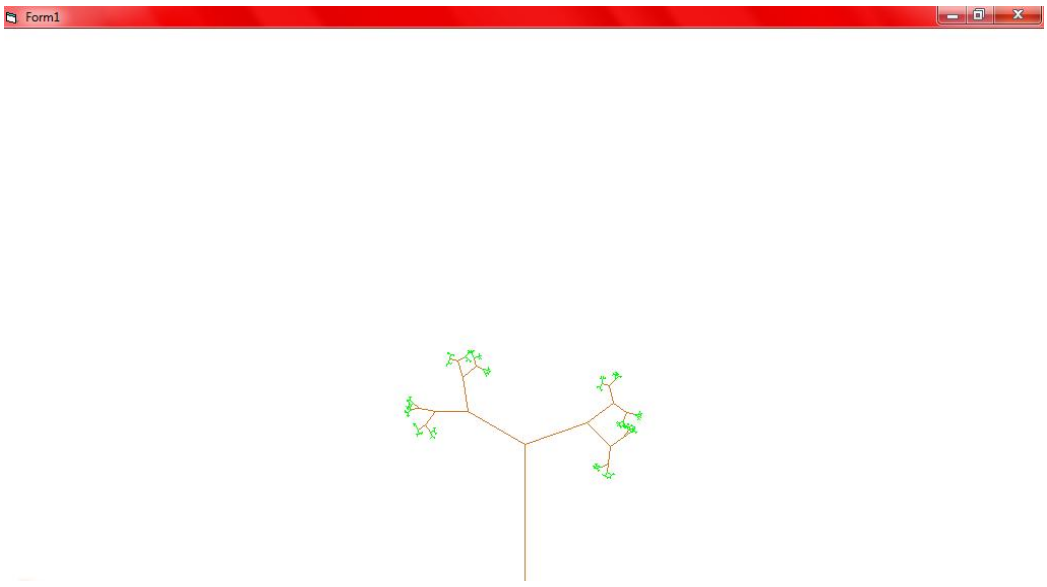
3.3.3 Codi del programa

```
Option Explicit
Const N = 10

Const P = 0.5
Const PI = 3.141592
Dim NARBRE As Long

Private Sub dibuixasubarbre(Ax As Double, Ay As Double, L As Double, Alpha As Double, nivell
As Integer, infectat As Integer)
Dim Vx As Double
Dim Vy As Double
Dim anglealeat As Double
Dim densitat As Integer

If nivell = N Then Exit Sub
For densitat = 0 To 3
anglealeat = Alpha + PI + PI / 2 + densitat * PI / 4 + Rnd() * (PI / 4)
Vx = L * P * Cos(anglealeat)
Vy = L * P * Sin(anglealeat)
Line (Ax, Ay)-(Ax + Vx, Ay + Vy), vbGreen
Call dibuixasubarbre(Ax + Vx, Ay + Vy, L * P, anglealeat, nivell + 1, 0)
Next

End Sub
```

```
Private Sub Command2_Click()
Dim Px As Double
Dim Qx As Double
Dim Py As Double
Dim Qy As Double
Dim Angle As Double
Dim Longitud As Double

Scale (-4, 4)-(4, 0)
Cls
Randomize Timer
Px = 0
Py = 0
Longitud = 2
Angle = PI / 2
Qx = Px + Longitud * Cos(Angle)
Qy = Py + Longitud * Sin(Angle)
Line (Px, Qx)-(Py, Qy), vbGreen
Call dibuixasubarbre(Py, Qy, Longitud, Angle, 1, 1)
End Sub
```

3.3.4 Resultats

Finalment, l'arbre que ha quedat, és el mateix però només amb aquest canvi de l'angle aleatori, així poden sortir tot tipus d'arbres diferents.



3.4 Infecció de l'arbre

3.4.1 Programa en un pas

Podem modificar l'anterior programa per que algunes branques surtin infectades, en color vermell. Estudiem primer una manera més senzilla d'implementar-ho així en el programa. A mesura que anem creant l'arbre, podem determinar si la branca que estem dibuixant mitjançant un experiment aleatori, per escollir el color verd o vermell.

Fem servir un paràmetre d'infecció que podem fixar al principi del programa amb un valor entre 0 i 1.

L'experiment aleatori (en realitat pseudoaleatori) consisteix en agafar el valor donat per la funció Rnd(). Si aquest valor és menor que Infecció, la branca sortirà infectada, i si es al contrari, sortirà sana.

Per exemple, si decidim un valor de 0.75 (= 75%) per la constant Infecció, els intervals de decisió seran els següents, segons el valor que resulti de la funció Rnd()

[0%, 75%] Hi ha infecció

(75%, 100] No hi ha infecció

Abans, però, hem de verificar si la branca mare esta infectada. No tindria sentit que la branca actual sortís infectada sense que ho estigui la branca mare. Per aquest motiu, hem de fer una modificació a la funció recursiva arbre i afegir un paràmetre que anomenarem Infectat, per controlar en cada moment si la branca anterior esta infectada o no.

El codi de la funció quedaria modificat de la següent manera

```
If aleatori > infecció Then
        Line (Ax, Ay) – (Ax+Vx, Ay+Vy), VbGreen
Else
        Line (Ax, Ay) – (Ax+Vx, Ay+Vy), VbRed
End If
```

Defectes

En executar el programa, observem que el seu funcionament no es del tot satisfactori. No tenim una visió nítida de les branques infectades, les branques vermelles, ja que en generar l'arbre a vegades dibuixa branques verdes a sobre d'aquelles.

La solució passa per dibuixar primer l'arbre en color verd i després anar marcant de color vermells la infecció.

3.4.2 Programa en dos passos

Generar l'arbre i guardar-lo en memòria

Hem de dibuixar l'arbre una primera vegada i després, aquest mateix arbre, l'hem de recórrer una altra vegada per dibuixar de color vermell les branques infectades. Això complica la part de programació, ja que ens obliga a guardar l'arbre en memòria.

Per guardar-lo farem servir una taula.

Quantes branques te un arbre en total?

Si de cada branca surten quatre, a mesura que avança el nivell n, s'aniran generant les següents branques

1, 4, 16, 64, ….., 4^n

Es una progressió geomètrica de Raó 4, i tenim una formula per sumar els termes. Es aquesta:

$$Sn = \frac{r \cdot a_n - a_1}{r - 1}$$

Fem una taula pels diferents valors de n

N	a_n	s_n
1	1	1
2	4	5
3	16	21
4	64	85
5	256	341
6	1024	1365
7	4096	5461
8	16384	21845
9	65536	87381
10	262144	349525
11	1048576	1398101
12	4194304	5592405

Els valor màxim de n que fem servir als nostres programes serà de 11 per tant agafarem un valor suficient per la dimensió de la taula de 1.400.000

Quina informació caldrà guardar de l'arbre per que es pugui tornar a dibuixar després ?

Per dibuixar la branca caldrà el punt inicial i el punt final, que son quatre valors reals (Double). Així ho desarem a la taula Coordenades de 4 columnes per 1400000 files.

També hem de guardar altra informació estructural de l'arbre

Nivell: Es el nivell de la branca

Mare: El programa necessita saber si la branca mare està infectada, per infectar o no a la branca actual

Infectat? Aquest paràmetre diu si una branca està infectada (1) o no (0)

Per tant,es crearà al principi del programa una taula Estructura de 3 columnes per 1400000 files.

Programa definitiu

Per millorar l'aspecte i la funcionalitat del programa hem afegit tres controls HSrollBar (barres de desplaçament) per poder donar valor a les variables **N** (nivell de branques que es dibuixaran), **P** (raó entre branques i sub branques) i **Infecció** (paràmetre d'infecció)

En un llenguatge de programació visual, com el Visual Basic, hem de programar que canviïn els valors de les variables quan es faci un canvi en aquests objectes

```
Private Sub HScrollInfeccio_Change()
LabelInf.Caption = "Infecció " & Str(HScrollInfeccio.Value) & " %"
Infeccio = HScrollInfeccio.Value / 100
End Sub
```

```
Private Sub HScrollN_Change()
LabelN.Caption = "N= " & HScrollN.Value
N = HScrollN.Value
End Sub
```

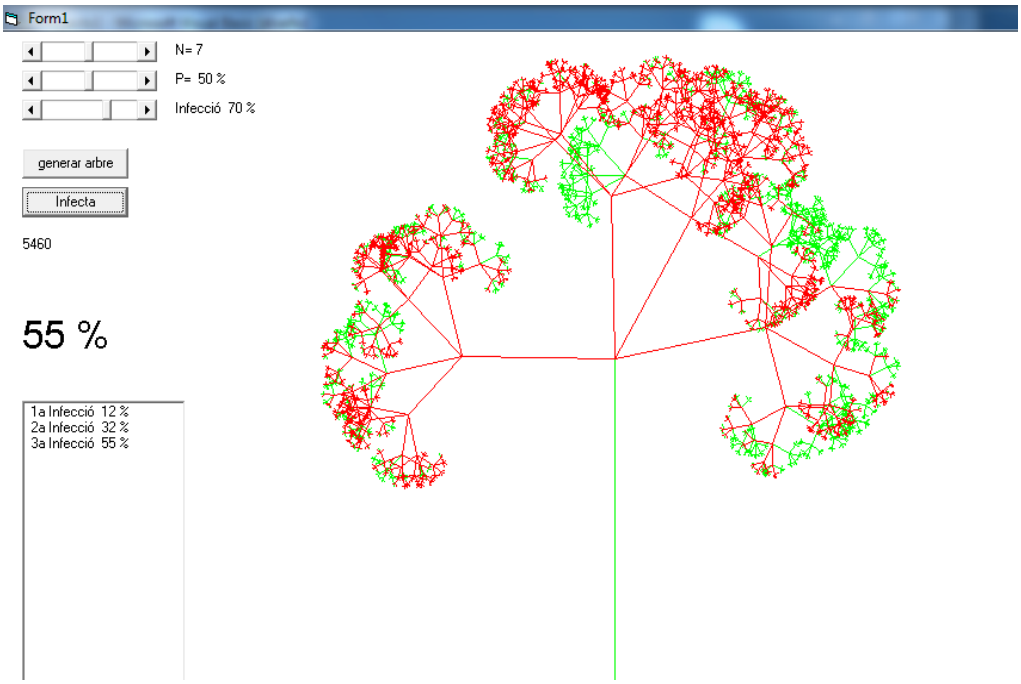
```
Private Sub HScrollP_Change()
LabelP.Caption = "P= " & Str(HScrollP.Value) & " %"
P = HScrollP.Value / 100
End Sub
```

Les etiquetes (label) són uns objectes que fem servir per visualitzar les dades a la pantalla, al costat de les barres de desplaçament.

Una altra modificació que hem fet al programa és la col·locació dels botons de comandament .

Generar Arbre : Dibuixa l'arbre amb els paràmetres escollits anteriorment. El dibuix és en color verd.

Infecta : Cada vegada que es fa clic, provoca una infecció. Si es pren varies vegades seguides la infecció es va acumulant, com es pot veure reflectit al label i al quadre que hi ha més avall.



Codi del programa

```
Option Explicit
Dim N

Dim P
Const PI = 3.141592
Dim Infeccio
Const NBranques = 3
Dim Tabla1(0 To 1500000, 3) As Long
Dim Tabla2(0 To 1500000, 4) As Double
Dim NARBRE As Long
Dim NumInfeccions

Private Sub dibuixasubarbre(Ax As Double, Ay As Double, L As Double, Alpha As
Double, nivell As Integer, infectat As Integer)
Dim Vx As Double
Dim Vy As Double
Dim anglealeat As Double
Dim densitat As Integer

Dim PARE As Long
If nivell = N Then Exit Sub
PARE = NARBRE
For densitat = 0 To NBranques
anglealeat = Alpha + PI + PI / 2 + densitat * PI / 4 + Rnd() * (PI / 4)
Vx = L * P * Cos(anglealeat)
Vy = L * P * Sin(anglealeat)
Line (Ax, Ay)-(Ax + Vx, Ay + Vy), vbGreen
NARBRE = NARBRE + 1
Tabla1(NARBRE, 1) = nivell
Tabla2(NARBRE, 1) = Ax
Tabla2(NARBRE, 2) = Ay
Tabla2(NARBRE, 3) = Ax + Vx
Tabla2(NARBRE, 4) = Ay + Vy
Tabla1(NARBRE, 2) = PARE
Tabla1(NARBRE, 3) = 0
Call dibuixasubarbre(Ax + Vx, Ay + Vy, L * P, anglealeat, nivell + 1, 0)
Next
End Sub
```

```
Private Sub Command1_Click()
Dim arbrenivell As Long
Dim z As Long
Dim index As Long
Dim aleat As Double
Dim PartInf As Integer
NumInfeccions = NumInfeccions + 1
PartInf = 0
For arbrenivell = 1 To N
For z = 1 To NARBRE
   index = Tabla1(z, 2)
   If (Tabla1(z, 1) = arbrenivell) And (Tabla1(index, 3) = 1) Then
      aleat = Rnd()
      If aleat < Infeccio Or Tabla1(z, 3) = 1 Then
      Line (Tabla2(z, 1), Tabla2(z, 2))-(Tabla2(z, 3), Tabla2(z, 4)), vbRed
      Tabla1(z, 3) = 1
      PartInf = PartInf + 1
      lblPartInf.Caption = Int(PartInf / NARBRE * 100) & " %"
      End If
   End If
Next z
Next arbrenivell
List1.AddItem Str(NumInfeccions) & "a Infecció  " & Int(PartInf / NARBRE * 100) & " %"
End Sub

Private Sub Command2_Click()
Dim Px As Double
Dim Qx As Double
Dim Py As Double
Dim Qy As Double
Dim Angle As Double
Dim Longitud As Double

List1.Clear
NumInfeccions = 0
Scale (-4, 4)-(4, 0)
Cls
Randomize Timer
Px = 0
Py = 0
Longitud = 2
```

```
Angle = PI / 2
Qx = Px + Longitud * Cos(Angle)
Qy = Py + Longitud * Sin(Angle)
Line (Px, Qx)-(Py, Qy), vbGreen
NARBRE = 0
Tabla1(NARBRE, 1) = 1
Tabla2(NARBRE, 1) = Px
Tabla2(NARBRE, 2) = Py
Tabla2(NARBRE, 3) = Qx
Tabla2(NARBRE, 4) = Qy
Tabla1(NARBRE, 2) = NARBRE
Tabla1(NARBRE, 3) = 1


Call dibuixasubarbre(Py, Qy, Longitud, Angle, 1, 1)
Label1.Caption = NARBRE

End Sub

Private Sub Form_Load()
HScrollN.Value = 7
HScrollP.Value = 50
HScrollInfeccio.Value = 70

End Sub

Private Sub HScrollInfeccio_Change()
LabelInf.Caption = "Infecció " & Str(HScrollInfeccio.Value) & " %"
Infeccio = HScrollInfeccio.Value / 100
End Sub

Private Sub HScrollN_Change()
LabelN.Caption = "N= " & HScrollN.Value
N = HScrollN.Value
End Sub

Private Sub HScrollP_Change()
LabelP.Caption = "P= " & Str(HScrollP.Value) & " %"
P = HScrollP.Value / 100
End Sub
```

3.5 Aplicació del programa

Aquest programa ens pot servir per estudiar alguns comportaments fractals, com per exemple l' infecció d'un arbre per la contaminació de la terra o també per àmbits humans, com l' estudi d'un pulmó infectat a causa del fum del tabac.

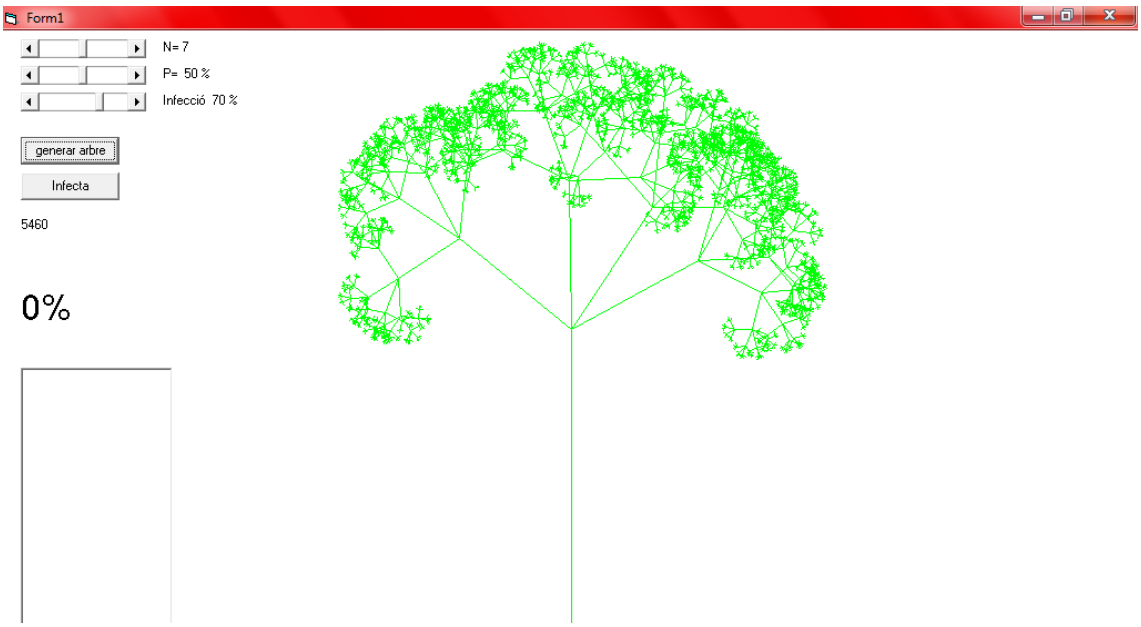
No és un programa en el que podrem esbrinar exactament que passarà, però si que ens podem fer una idea i si el millorem, potser pot ser útil per estudiar malalties infeccioses.

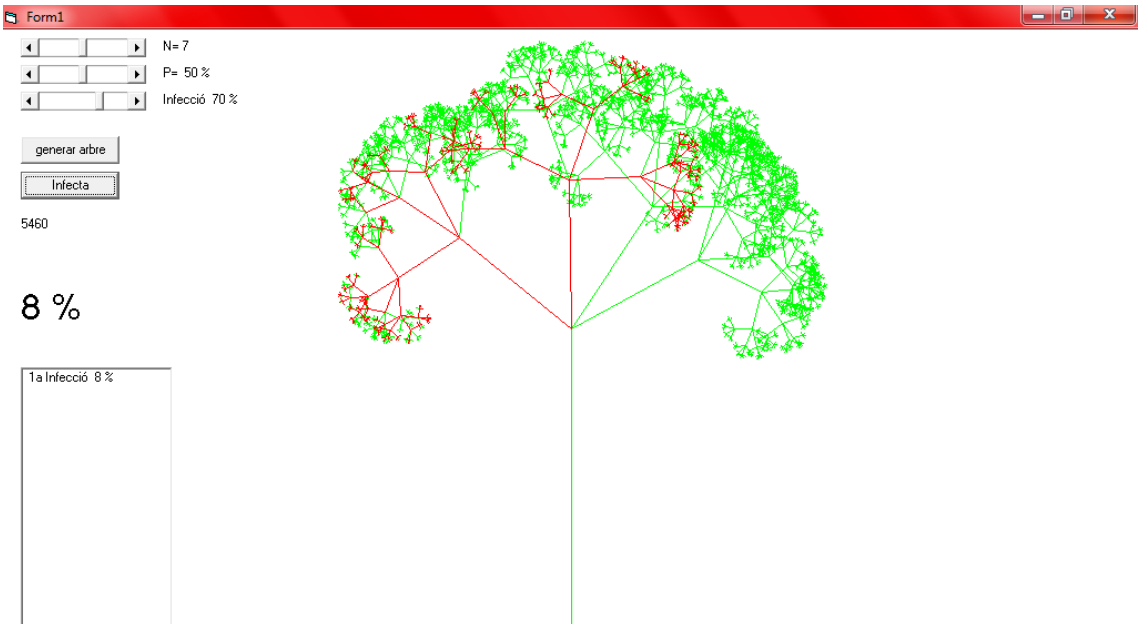
Per exemple, ens introduirem en una simulació de l' estudi d'un pulmó que a causa del tabac, cada any va infectant-se més.

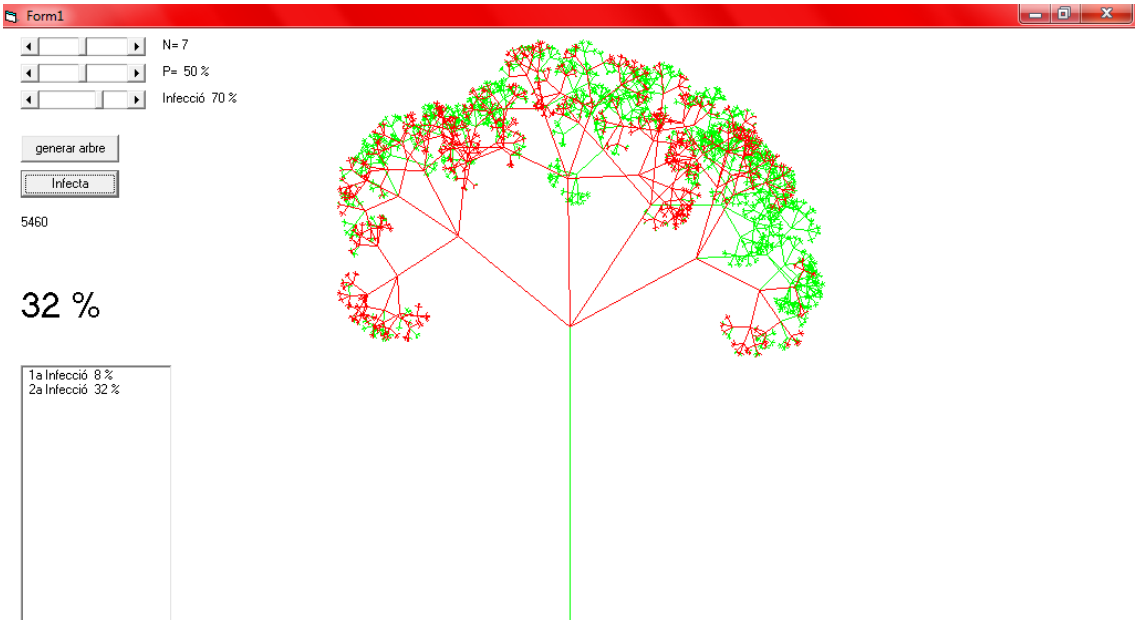
En aquest model, podríem esperar que l' infecció es produís de manera exponencial, és a dir, que després de cada any la zona infectada molt més gran que l'anterior, però en el meu model la infecció no es produeix d'una manera tan rapida. El creixement és més sostingut. Això podria ser degut a que les cèl·lules reaccionen contra l' infecció i es van adaptant i ofereixen una certa resistència que fa que el creixement no sigui tan ràpid.

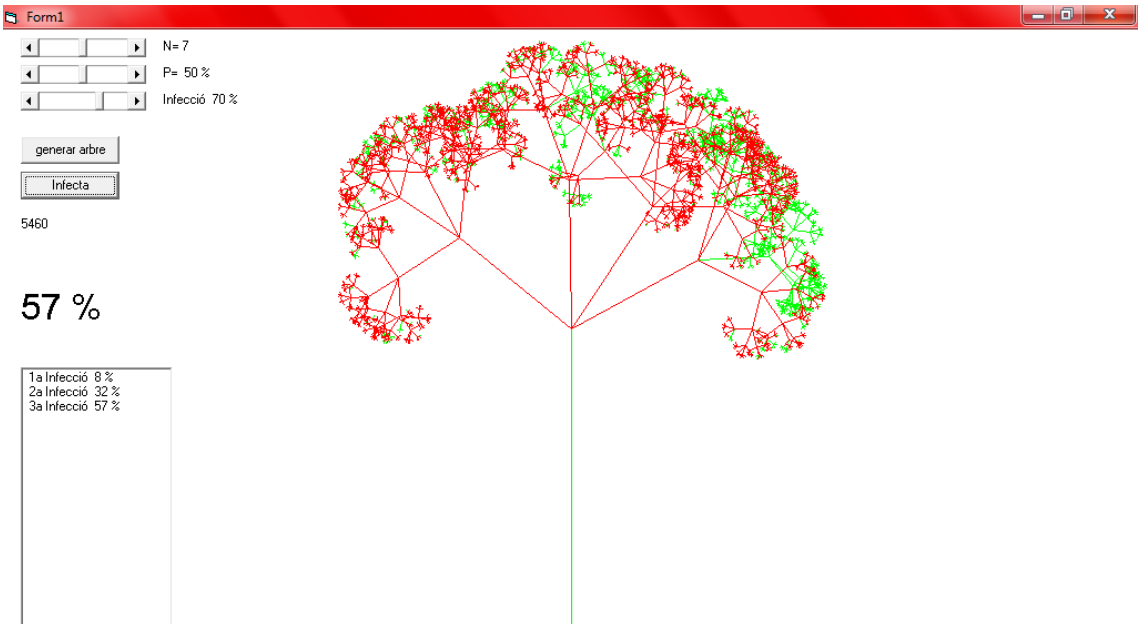
En el nostre programa, de moment, deixarem fixes les variables N=7 i P=50%, a més estudiant infeccions d'altres pulmons, he decidit deixar una infecció del 70%, però podríem canviar-ho al nostre gust.

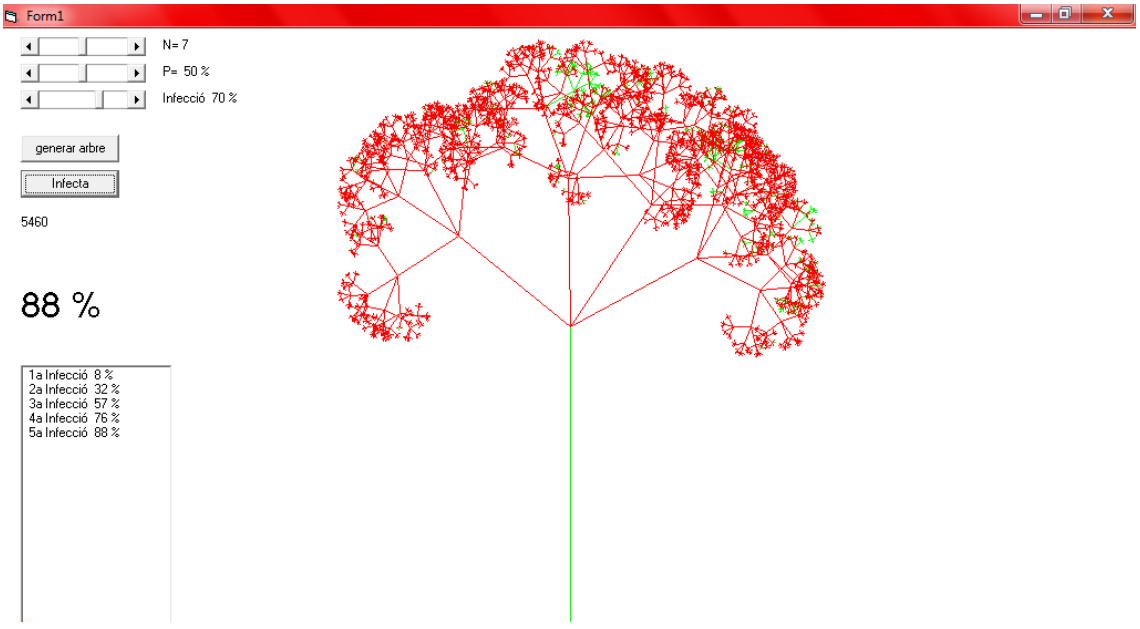
En les captures de pantalles següents es pot veure la infecció amb més claredat.

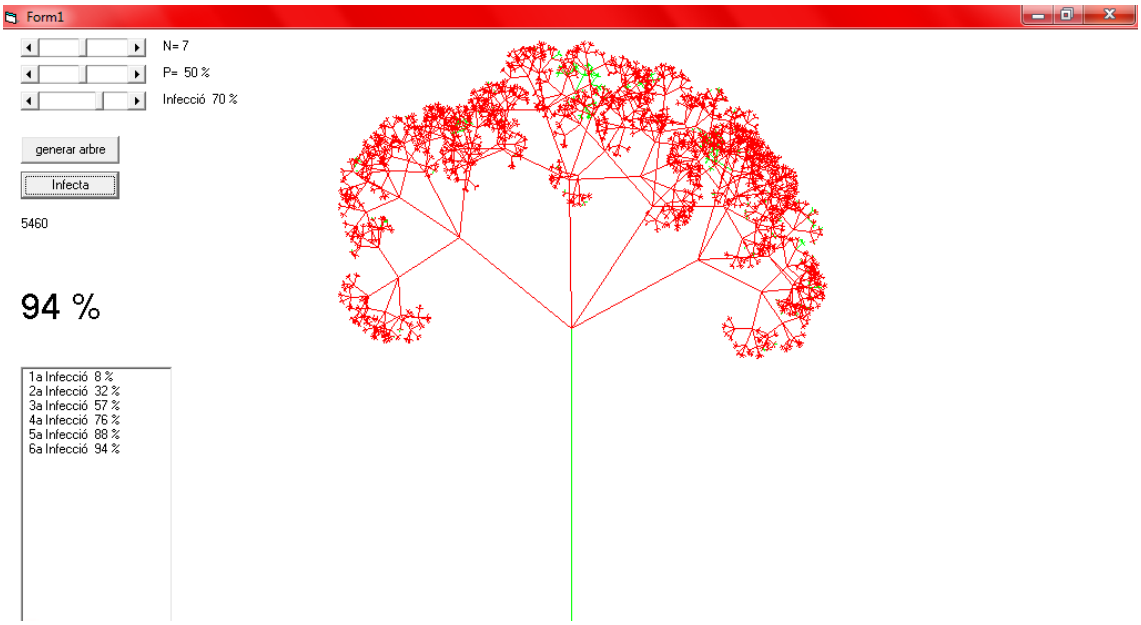


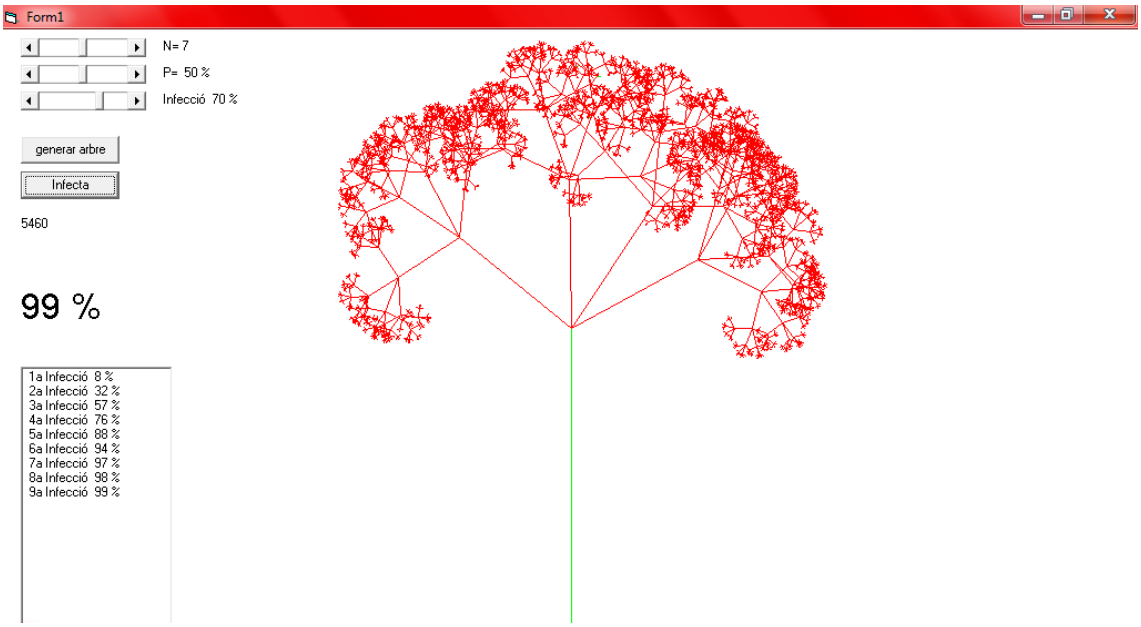

Form1
N= 7
P= 50 %
Infecció 70 %
generar arbre
Infecta
5460
8 %
1a Infecció 8 %

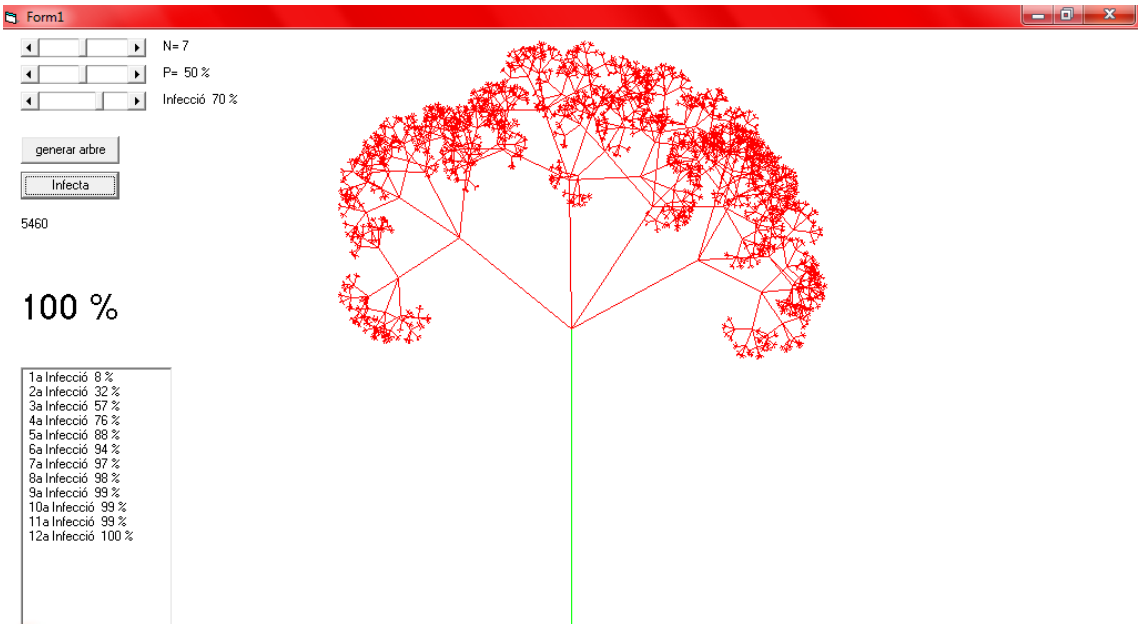

Form1
N= 7
P= 50 %
Infecció 70 %
generar arbre
Infecta
5460
32 %
1a Infecció 8 %
2a Infecció 32 %


Form1
N= 7
P= 50 %
Infecció 70 %
generar arbre
Infecta
5460
57 %
1a Infecció 8 %
2a Infecció 32 %
3a Infecció 57 %


Form1
N= 7
P= 50 %
Infecció 70 %
generar arbre
Infecta
5460
88 %
1a Infecció 8 %
2a Infecció 32 %
3a Infecció 57 %
4a Infecció 76 %
5a Infecció 88 %


Form1
N= 7
P= 50 %
Infecció 70 %
generar arbre
Infecta
5460
94 %
1a Infecció 8 %
2a Infecció 32 %
3a Infecció 57 %
4a Infecció 76 %
5a Infecció 88 %
6a Infecció 94 %


Form1
N= 7
P= 50 %
Infecció 70 %
generar arbre
Infecta
5460
99 %
1a Infecció 8 %
2a Infecció 32 %
3a Infecció 57 %
4a Infecció 76 %
5a Infecció 88 %
6a Infecció 94 %
7a Infecció 97 %
8a Infecció 98 %
9a Infecció 99 %



Ens plantegem si aquest model es pot fer servir per estudiar la infecció d'un pulmó d'una persona fumadora.

Els bronquis d'un pulmó es ramifiquen en els bronquíols de manera semblant a com ho fan les branques i les sub branques d'un arbre.

La infecció depèn d'uns paràmetres, el coeficient d'infecció que té valors 60 % fins a 95 % com es veu en la taula que hi ha més endavant, però també té una component aleatòria, això pot representar un pulmó més resistent o menys resistent.

Mantenint el valors de N i P que el programa presenta per defecte (N=7 i P=50%), centrarem la nostra atenció en el valor de la variable Infecció i anirem infectant el pulmó en diferents fases. Cada fase seria fer un clic al botó d'infecció, i podria representar que ha passat un període de temps (per exemple un any).

Obtenim aquesta taula de resultats El resultats són percentatge de infecció de l'arbre

	Coeficient d'infecció							
	60%	65%	70%	75%	80%	85%	90%	95%
1a inf	6	3	9	15	33	49	39	83
2a inf	19	15	30	49	66	85	78	98
3a inf	33	36	58	75	87	95	93	99
4a inf	51	58	74	88	96	99	97	99
5a inf	65	72	86	95	98	99	99	100

Cal tenir en compte que en altres execucions del programa poden sortir resultats una mica diferents. També en aquesta taula de prova veiem algun resultat distorsionat, degut al

component aleatori del model, però mirats el resultat de forma global, s'observen les tendències.

Si el que es tracta és de simular la infecció del pulmó, podríem concloure que potser el creixement no es correspon ben bé al comportament real. El programa ens dona un creixement sostingut, sobretot en les últimes fases. Segurament aquest creixement s'hauria d'accelerar a partir d'un determinat moment o paulatinament, és a dir, poc a poc. Per tant, una possible millora del model seria tenir en compte aquesta acceleració.

4. Contingut DVD

Amb el treball s'adjunta un DVD amb les següents carpetes:

- Mandelbrot blanc i negre
- Mandelbrot en colors
- Arbre simètric de dues branques
- Arbre amb branques a l'atzar
- Arbre amb infecció en un pas
- Arbre amb infecció en dos passos

Dintre de cada carpeta hi ha el codi font i el programa executable.

5. Conclusió

En realitzar aquest treball m'he adonat que totes les coses que estudiem (en aquest cas dins de les matemàtiques) serveixen per alguna cosa, tot té la seva utilitat.

He vist que els nombres complexos, que en realitat són una invenció dels matemàtics, poden tenir utilitats diverses. Per exemple podem mirar-los des del punt de vista de l'art.. Sense anar més lluny, els fractals que fem referència al treball donen lloc a formes molt maques.

També he pogut comprovar que, tot i que són programes curts els que donen lloc al dibuix de Mandelbrot o al dibuix dels arbres, són molt difícils d'escriure. A vegades, pot resultar molt complicat traslladar les idees que comprenem i que més o menys tenim al cap a l'escriptura de les instruccions d'un programa.

He après que no tots els llenguatges de programació serveixen igual. Per exemple, l' Scratch que pot ser un llenguatge senzill per aprendre i fer coses fàcils (fins i tot ho fan servir els nens!), no ha funcionat massa bé per fer el dibuix de Mandelbrot.

Per últim, he experimentat com es fa una simulació d'un problema de la nostra vida, la infecció d'un pulmó o un d'un arbre. He comprovat que després de la dificultat d'escriure el programa, encara no s'adaptava a la situació real. O sigui que aquest programa encara el podríem millorar si fem un estudi més acurat del tema. Caldria afegir un estudi mèdic per tal de calibrar els diferents paràmetres que surten al programa i modificar el funcionament en alguna part. Per exemple, quan vam observar que la infecció hauria de tenir un creixement exponencial.

6. Bibliografia

Wikipedia:

- http://ca.wikipedia.org/wiki/Nombre_complex (Abril 2014)
- http://ca.wikipedia.org/wiki/Fractal (Abril 2014)
- http://en.wikipedia.org/wiki/Mandelbrot_set (Maig 2014)
- http://ca.wikipedia.org/wiki/Triangle_de_Sierpi%C5%84ski (Novembre 2014)

Fractfinder:

- http://www.fractfinder.es/ (Abril 2014)
- http://www.fractfinder.es/laboratorio/index.php (Abril 2014)

Altres webs i llibres de cerca:

- http://comvisualfractales.blogspot.com.es/ (Octubre 2014)
- Francisco Javier Ceballos Sierra , Visual Basic Curso de programación , Editorial Ra-Ma, 1999.
- T.M Apostol, Análisis Matemático, Editorial Reverté S.A, 1982.

Imatges:

- http://upload.wikimedia.org/math/5/f/3/5f35e22e728ceda32f443ca18acb0e62.png
- http://upload.wikimedia.org/wikipedia/commons/2/21/Mandel_zoom_00_mandelbrot_set.jpg
- http://math.bu.edu/DYSYS/chaos-game/sierp-det.GIF
- http://www.fractfinder.es/ (4 Imatges de Mandelbrot)