Dove andremo a finire?

I.I.S. "Vittorio Veneto" Città della Vittoria – Vittorio Veneto (TV)
Classe: 4°B I.T.I.S.
Insegnante di riferimento: Clara Della Pietà
Ricercatore: Federico Totaro
Ragazzi partecipanti: Andrea Azzalini, Filippo Casagrande, Stefano Da Ros, Enrico Dal Ben, Luca Dal Cin, Patrick Dal Pio Luogo, Alberto Dall'Arche, Nicola De Biasi, Marco Luca Dori, Dragos Andrei Maxineau, Renato Meneghin, Francesco Polese, Marco Russo, Simone Selvestrel

L'avventura Math.en.Jeans ci ha portato a indagare attorno ai quattro problemi che il nostro *mentore* Federico Totaro ci ha proposto quando ci siamo visti a Rho, nell'ormai lontano autunno 2012. Gli studenti si sono fatti piacevolmente coinvolgere dalle attività proposte e al nostro ritorno a Vittorio Veneto ci siamo messi all'opera.
La modalità di lavoro scelta è stata quella del lavoro di gruppo e il lavoro è stato svolto interamente in orario curriculare. I gruppi (due gruppi da tre e due gruppi da quattro studenti) hanno lavorato in laboratorio di informatica due ore alla settimana per circa quattro mesi.
La scelta di lavorare utilizzando un supporto informatico è stata quasi scontata data la tipicità dell'indirizzo di studio (istituto tecnico indirizzo informatico) e, ad eccezione forse per alcuni aspetti relativi al "Life", l'uso del calcolatore e della rete internet non ha tolto spontaneità e genuinità al lavoro. Fondamentale l'utilizzo della piattaforma Moodle per i contatti con Federico Totaro.
Come insegnante ho cercato di non influenzare l'attività di ricerca degli studenti, ho soprattutto cercato di far sì che tutti potessero apportare il proprio contributo e ho sollecitato ad andare avanti, ponendo qualche interrogativo nei momenti in cui qualche gruppo pensava di aver finito il lavoro. Su consiglio di Federico non ho introdotto nuovi argomenti; così, nel corso di tutta l'attività, non si è mai fatto cenno allo studio dei sistemi dinamici discreti (dei quali i problemi proposti altro non erano che quattro esempi). Soltanto al termine delle presentazioni dei risultati di ciascun gruppo al resto della classe, è stato presentato un articolo che ho ritenuto alla portata degli studenti e che si prestava bene a fare da sintesi e che permetteva di sistematizzare il lavoro svolto dai gruppi (http://www.matematica.it/impedovo/articoli/Sistemi%20dinamici%20discreti.pdf). Un solo gruppo ha rivisto la presentazione dopo la presa visione dell'articolo, introducendo qualche termine proprio del contesto.

Pari o Dispari? (Andrea Azzalini, Filippo Casagrande, Stefano Da Ros)

Prendi un numero intero e positivo, poi:

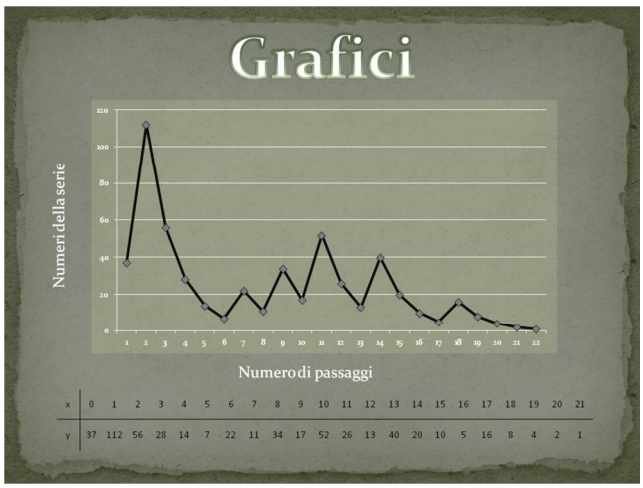
- *se è pari, dividilo per due;*
- *se invece è dispari, moltiplicalo per tre e aggiungi uno.*

Che numero hai ottenuto? Ricomincia da capo con il numero trovato (guarda se è pari o dispari...). Hai sempre numeri nuovi, o talvolta capiti su numeri già visti?

Il problema è stato da subito formalizzato.
Il sistema dinamico che propone il problema è il seguente:

$$x_{n+1} = \begin{cases} x_n/2 & \text{se } x_n \text{ è pari;} \\ x_n \cdot 3 + 1 & \text{se } x_n \text{ è dispari.} \end{cases}$$

Innanzitutto abbiamo analizzato il problema su carta. Dopo svariati tentativi con numeri casuali ci siamo accorti che tutte le successioni numeriche analizzate generavano una stessa sequenza di numeri che poi continuava a ripetersi all'infinito: 4, 2, 1.
Successivamente abbiamo implementato il problema con i mezzi informatici a nostra disposizione e continuato ad analizzare diversi numeri per poi metterli in relazione al numero di passaggi effettuati prima della conclusione della serie con il *loop*, creando anche alcuni grafici in excel.
L'utilizzo del computer ci ha permesso di effettuare un notevole numero di prove; inoltre per ciascuna prova è stato anche contato il numero di passaggi eseguiti prima di incontrare la fatidica sequenza 4, 2, 1.



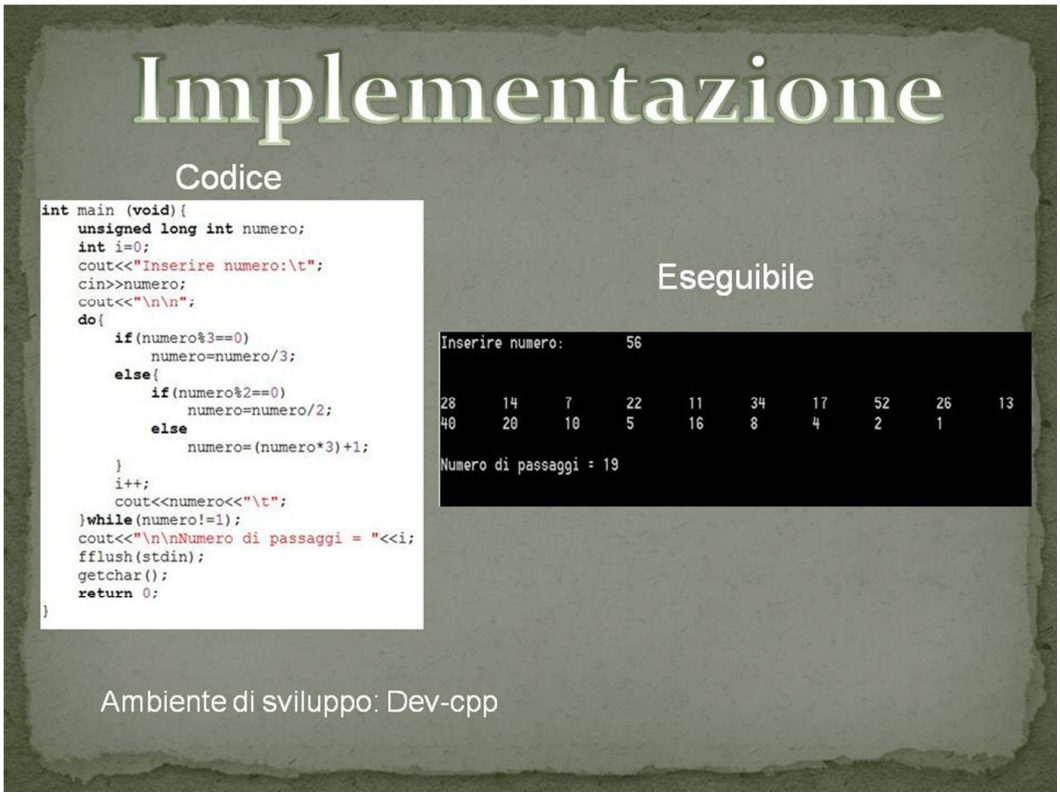
Abbiamo poi provato a cambiare le regole del problema nel modo seguente. Dopo aver preso un numero intero e positivo:

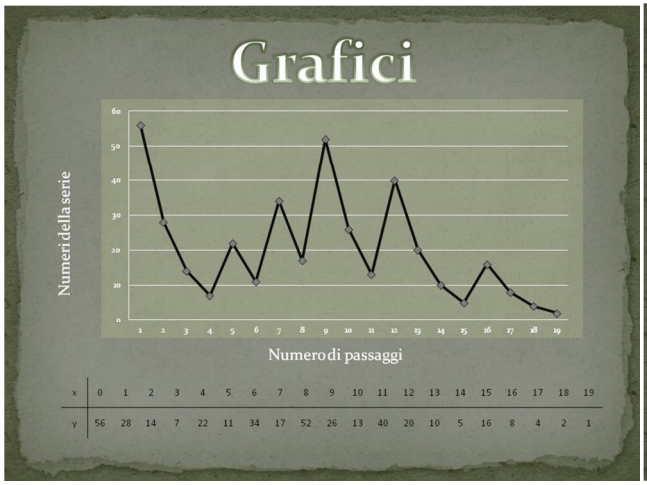
- se è pari, dividilo per due;
- se è divisibile per tre, dividilo per tre;
- altrimenti se è dispari e non divisibile per tre, moltiplicalo per tre ed aggiungi uno.

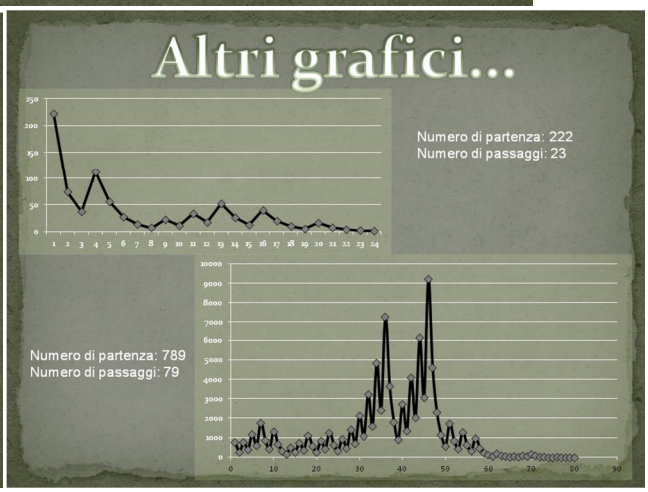
Il nostro nuovo sistema dinamico diventa quindi:

$$x_{n+1} = \begin{cases} x_n/2 & \text{se } x_n \text{ è pari;} \\ x_n/3 & \text{se } x_n \text{ è divisibile per 3;} \\ x_n \cdot 3 + 1 & \text{se } x_n \text{ è dispari e non divisibile per 3.} \end{cases}$$

Anche in questo caso si può notare che la successione si conclude nello stesso modo dell'altra.



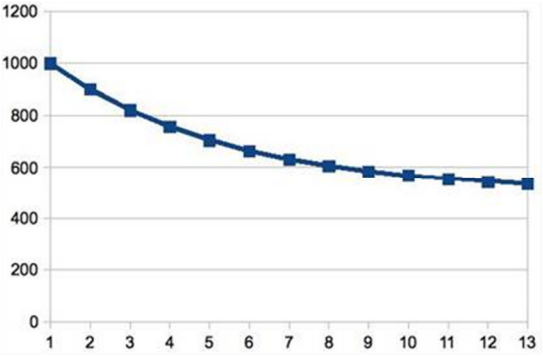




Per facilitare la ricerca del valore in corrispondenza del quale si effettuano più passaggi abbiamo implementato un nuovo algoritmo che fa dei test con numeri consecutivi e restituisce quello che ha più passaggi. Da qui abbiamo evinto che alcuni gruppi di numeri consecutivi hanno lo stesso numero di passaggi.

Impianto di pesca sportiva / Conto in banca (Enrico Dal Ben, Luca Dal Cin, Patrick Dal Pio Luogo, Alberto Dall'Arche)

Il gestore di un laghetto di pesca sportiva si ritrova a fare due conti prima dell'inizio della nuova stagione. Una particolare legge gli impone di limitare mensilmente la pesca al 20% degli esemplari presenti in acqua, limite che raggiunge puntualmente grazie all'alto numero di pescatori che frequentano il suo impianto. All'inizio dell'anno precedente il bacino conteneva 1000 trote. Per arginare la diminuzione causata dalla pesca, il gestore introduceva alla fine di ogni mese 100 nuovi esemplari. Nonostante il suo intervento, l'uomo scopre con disappunto che, alla fine dell'anno, il numero di trote presenti nel suo bacino si è ridotto quasi della metà del numero iniziale.



Come se non bastasse, l'inizio della nuova stagione ha portato all'uomo la concorrenza di un secondo impianto di pesca, sorto a pochi metri di distanza dal suo, che a dire il vero è un poco più piccolo: il laghetto contiene infatti, all'inizio, solo 250 trote. Quasi a volerlo copiare, anche il nuovo arrivato rimpingua ogni mese il proprio bacino con esattamente 100 esemplari di trote.

Per entrambi gli impianti continua a vigere il limite mensile del 20% di pesci pescati, ed entrambi hanno clienti a sufficienza per raggiungerlo. Tuttavia, mentre il primo impianto si ritrova ad avere ogni mese sempre meno trote, nel secondo impianto accade che il numero di pesci aumenta costantemente. Le condizioni climatiche, come quelle biologiche, sono le stesse per entrambi gli impianti. Che cosa sta succedendo? Come potrebbe l'uomo correre ai ripari?

Altro problema. Pensate a un conto corrente, sul quale ogni mese vengono effettuati dei versamenti fissi (ad es. uno stipendio) e dei prelievi percentuali (ad es. una tassa).

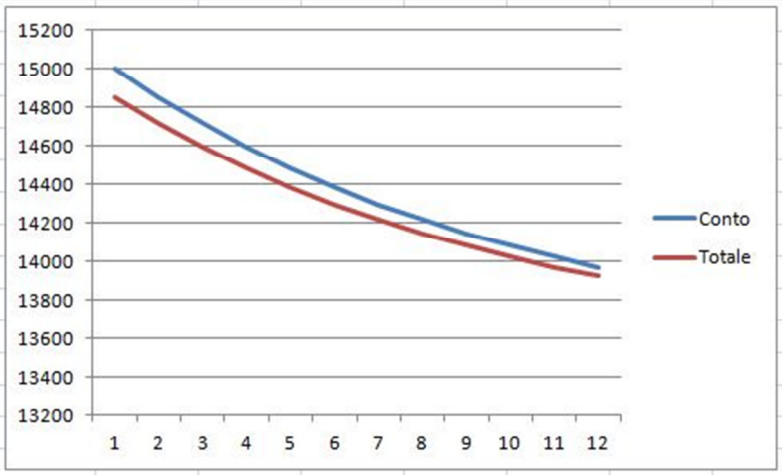
Si tratta di due problemi molto simili.
Con l'aiuto delle nostre conoscenze informatiche e matematiche ci siamo mossi e abbiamo costruito dei programmi utilizzando due ambienti di lavoro diversi per ottenere risultati più precisi: Microsoft Excel e Visual C#.
Con Excel abbiamo fatto un primo test, con l'ausilio di una tabella e di un grafico, per cercare di capire l'andamento che prendevano il fenomeno dei due impianti di pesca e del conto corrente.
Con Visual C#, invece, abbiamo creato un programma che permette di osservare il modificarsi del numero di pesci nel laghetto.
Abbiamo così concluso che i due problemi sono analoghi. Sia il numero di pesci sia l'ammontare del conto corrente, infatti, tendono al limite della successione:

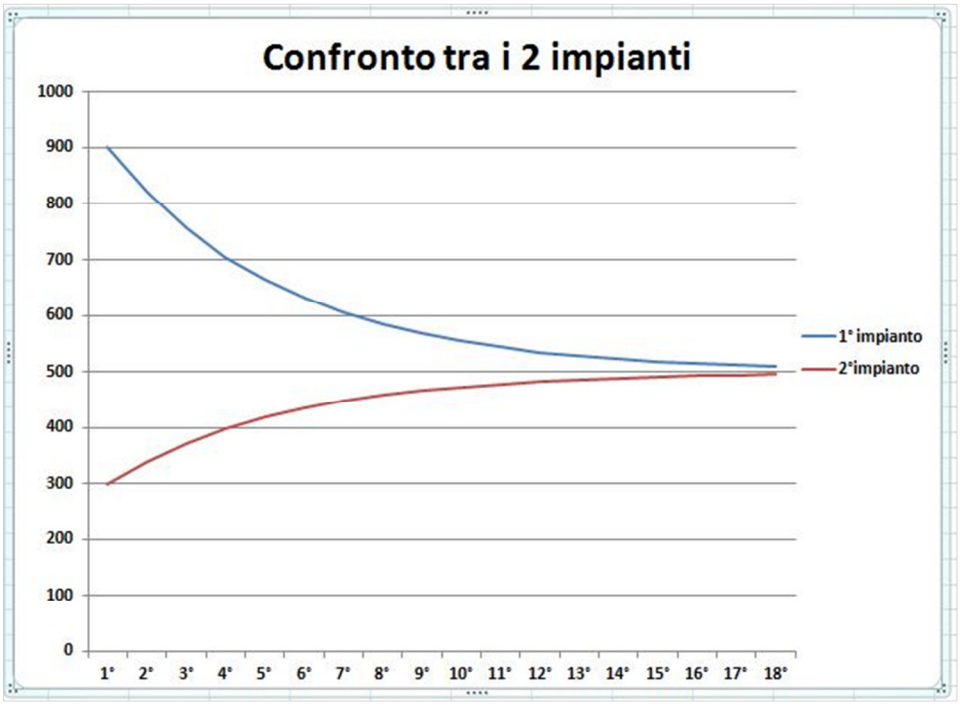
$$\begin{cases} f(0) = d \\ f(n+1) = f(n) \cdot (1-k) + c \end{cases}$$

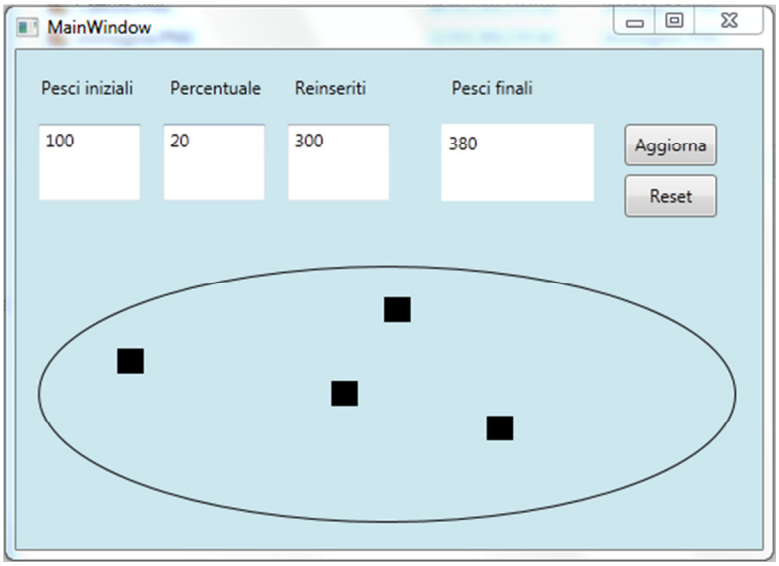
dove *d* è il numero dei pesci/capitale al tempo zero, *k* la percentuale, *c* i pesci reinseriti/versamento fisso e *n* l'istante di tempo.

Problema del conto corrente

Conto	15000
Tasse	10
Stipendio	1500







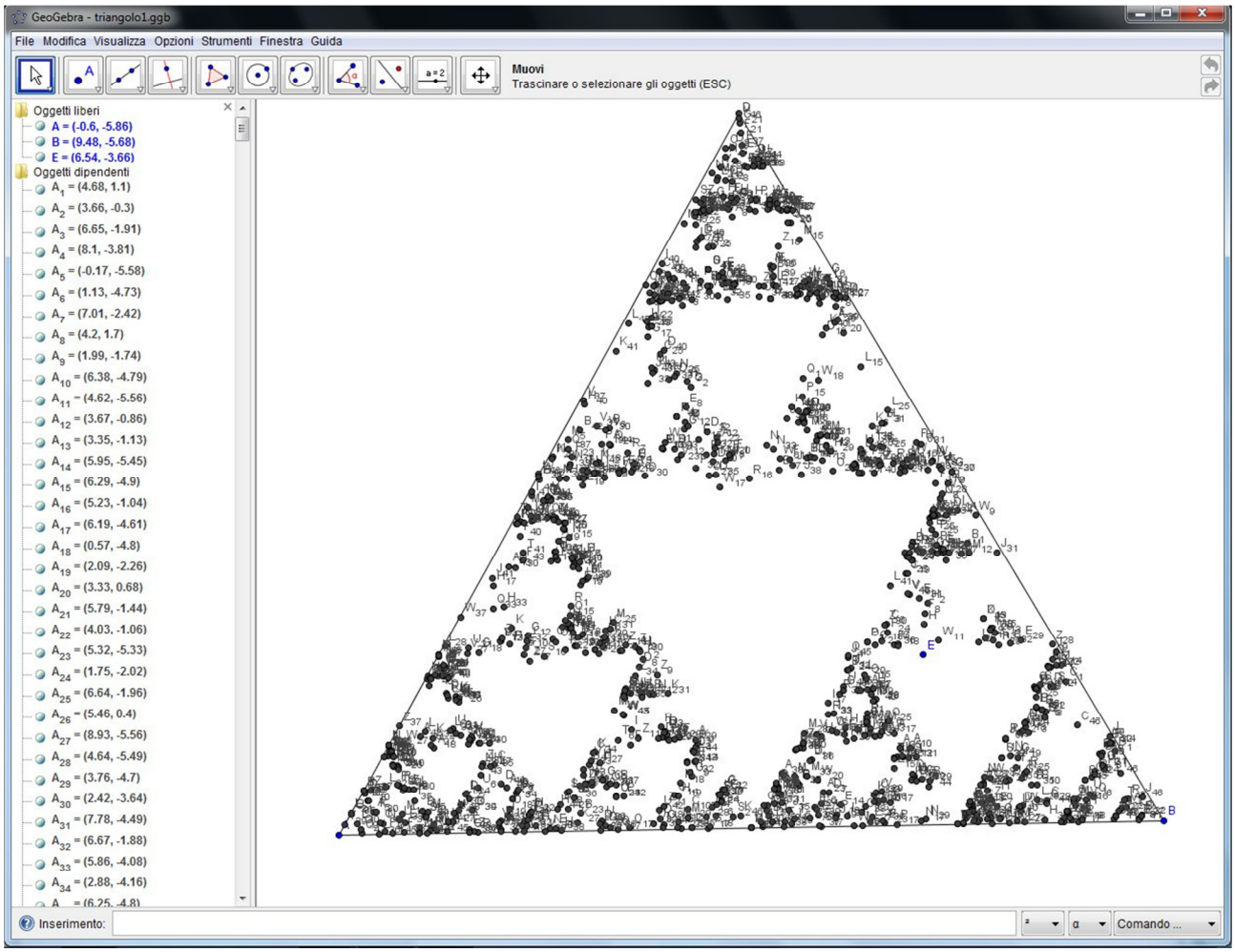
Punti medi (Nicola De Biasi, Marco Luca Dori, Dragos Andrei Maxineanu)

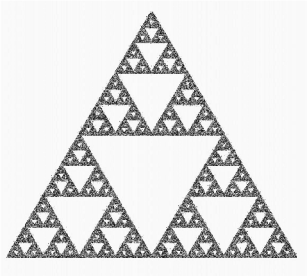
Prendi un triangolo equilatero e rinomina i suoi vertici: vertice A, vertice B, vertice C.
Con una matita, scegli a piacere un punto all'interno di questo triangolo, poi prendi un dado.
Stabiliamo una regola:

- *se tirando il dado esce 1 o 2 sceglieremo il vertice A;*
- *se esce 3 oppure 4 sceglieremo il vertice B;*
- *se esce 5 oppure 6 sceglieremo il vertice C.*

Tira il dado, e partendo dal punto che hai scelto all'inizio, muoviti in linea retta verso il vertice che ti ha indicato il dado, fermandoti a metà strada. Segna il punto che hai trovato con una matita. Ripeti poi la procedura, tutte le volte tirando il dado e andando a segnare il punto medio del segmento che ha per estremi il punto trovato al passo precedente e il vertice indicato del dado.
Immaginando di poter andare avanti a farlo tantissime volte (ed essendo sempre sufficientemente precisi), si riuscirà ad annerire il triangolo per intero, punto per punto? Cambiando figura il risultato è lo stesso? E cambiando regola?

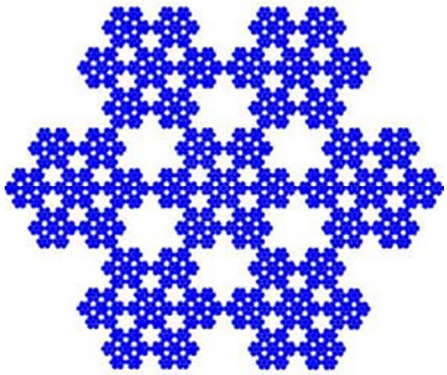
Questo gruppo ha lavorato utilizzando dapprima GeoGebra (al posto della matita) per capire che tipo di evoluzione prendeva la figura.

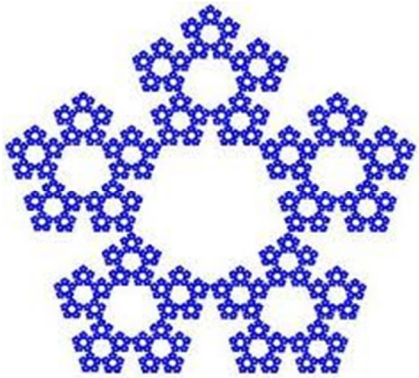




Utilizzando poi Small Basic e Visual C# è stato implementato un algoritmo che permette di visualizzare la figura ottenuta a partire da un qualunque punto interno al triangolo. In questo modo è stato possibile verificare che si ottiene sempre una figura molto simile al triangolo di Sierpinski.

Successivamente sono state modificate le regole di costruzione: anziché considerare il punto medio è stato tracciato il punto che dista 1/3 da vertice, oppure ¼. Le figure ottenute ricordavano sempre il triangolo di Sierspinski ma con spazi vuoti sempre maggiori. Altri tentativi sono stati fatti in seguito a partire da pentagoni ed esagoni al posto del triangolo, ottenendo immagini analoghe.

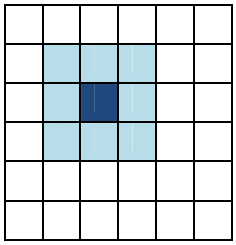




Uno studente (Marco Luca Dori) si è appassionato a questo tipo di costruzioni ed ha realizzato un programma che genera un albero binario particolare. I codici dei programmi realizzati sono riportati al termine della relazione.

Game of life (Renato Meneghin, Francesco Polese, Marco Russo, Simone Selvestrel)

Ogni cella della scacchiera (che immaginiamo essere indefinitivamente estesa) ammette due configurazioni: può essere vuota (cellula morta), oppure ospitare una cellula viva. Quando si parla di 'celle vicine', ci si riferisce sempre alle 8 caselle che circondano la cella di cui stiamo parlando (nel disegno, le celle vicine alla cella blu sono le 8 celle azzurre).



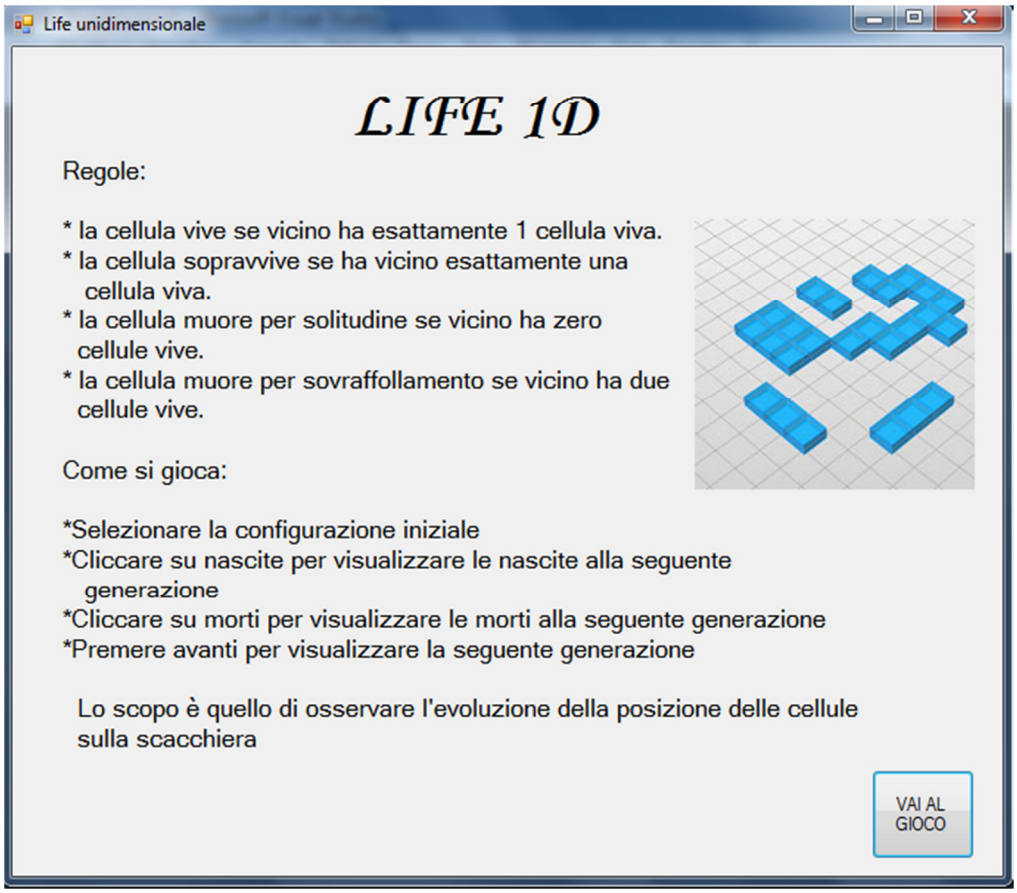
All'inizio, il giocatore prende un certo numero di pedine (le cellule), e decide come disporle sulla scacchiera. A questo punto il sistema cellulare evolverà step-by-step rispettando le regole seguenti:

- *se una cellula viva ha intorno a sé 2 o 3 cellule vive, la cellula rimane in vita;*
- *se una cellula viva è isolata o ha vicino solo 1 cellula viva, essa muore per solitudine. Se invece ha accanto a sé 4 o più cellule vive, essa muore per sovraffollamento. In entrambi i casi la pedina andrà rimossa dalla scacchiera;*
- *se una casella vuota ha intorno a sé ESATTAMENTE 3 cellule vive, in quella casella nasce una nuova cellula.*

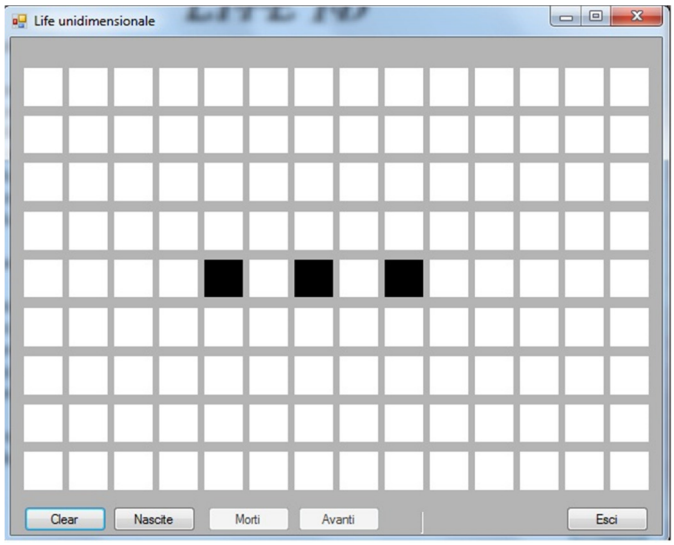
Questo gruppo è quello che all'inizio ha avuto un po' più di difficoltà, dovuta all'eccesso di informazioni. Infatti, in rete si trovano siti in cui si può giocare e vedere come evolvono le diverse configurazioni e sono facilmente reperibili articoli che in cui si illustrano alcune configurazioni particolari (alianti, etc). Su suggerimento di Federico, il lavoro si è spostato al caso *1D*.

Nella versione del life 1D il "vicinato" di ciascuna cella è costituito da sole due celle. In pratica le cellule sono tutte affiancate lungo una retta. La cella ospiterà vita se vicino ha esattamente 1 cellula viva (se la cella era vuota, sorgerà una nuova cellula); sarà vuota (morte) se vicino ha 0 oppure 2 cellule vive.

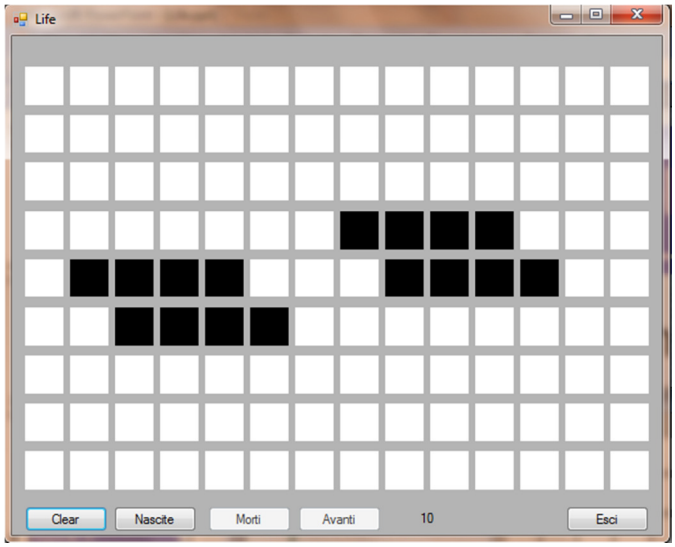
Il *Life* unidimensionale è, in teoria, un nastro che si estende indefinitamente nelle due direzioni. Quello da noi progettato (e da cui sono tratte le immagini qui riportate) per motivi di praticità a livello software è rappresentato in forma di matrice di dimensione 14X9, e va quindi pensato come un nastro ripiegato.
Di seguito un'immagine della schermata iniziale del programma da noi realizzato, il codice è riportato al termine di queste pagine.



Partendo dalla figura seguente

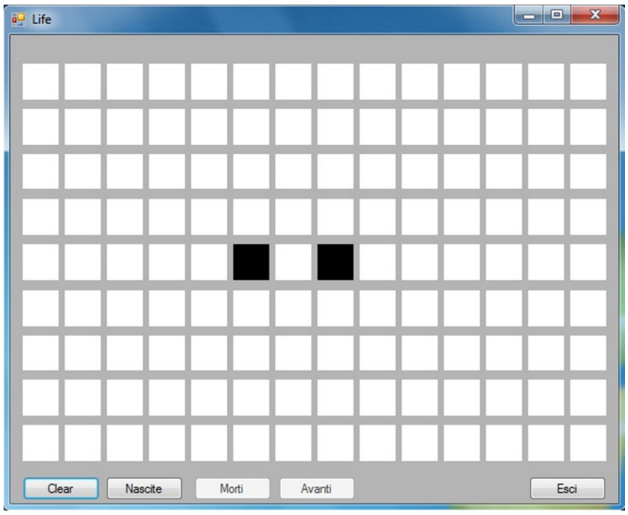


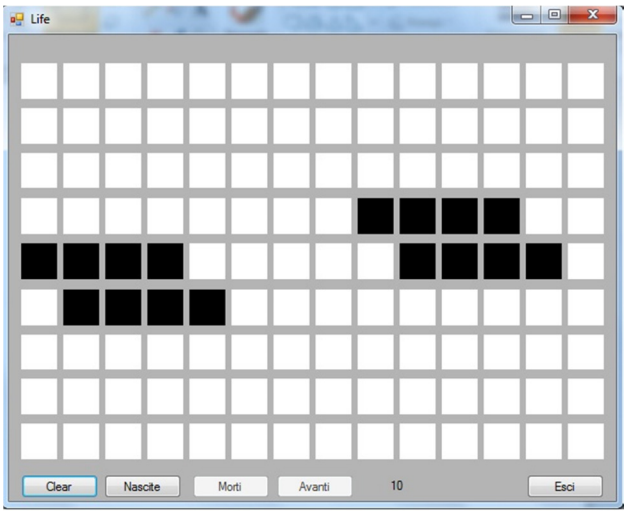
dopo dieci generazioni si ha questa configurazione:



La figura iniziale si espande e riproduce periodicamente la stessa sequenza di combinazioni, che si ripetono un numero sempre maggiore di volte.

Ed ecco l'evoluzione di un'altra configurazione:





Anche in questo caso la figura iniziale si espande e riproduce periodicamente la stessa sequenza di combinazioni, che si ripetono un numero sempre maggiore di volte.

Dopo aver provato diverse combinazioni, abbiamo capito che nel *Life 1D*, a differenza del *Life* classico, le combinazioni non si estinguono mai ma continuano ad espandersi all'infinito nelle due direzioni della retta.

Codice Triangolo 1/2

```
'disegna il triangolo
GraphicsWindow.Height=1014
GraphicsWindow.Width=1270
GraphicsWindow.Top=0
GraphicsWindow.Left=0
GraphicsWindow.DrawTriangle(300, 800, 900,800,600,300)

start = Controls.AddButton("Start",70,10)
reset = Controls.AddButton("Ricomincia",120,10)
exit = Controls.AddButton("Chiudi",10,10)

flag=1

'procedura per trovare il punto medio
Sub puntoMedio
  If c=1 Or c=2 Then
  x=Math.Abs(300+x)/2
  y=Math.Abs(800+y)/2
  setPoint()
  endif
  If c=3 Or c=4 Then
  x=Math.Abs(900+x)/2
  y=Math.Abs(800+y)/2
  setPoint()
  endif
  If c=5 Or c=6 Then
  x=Math.Abs(600+x)/2
  y=Math.Abs(300+y)/2
  setPoint()
  endif
EndSub

'procedura per disegnare il primo punto da mouse
  Sub primoPunto
    If Mouse.IsLeftButtonDown Then
```

```
      x=GraphicsWindow.MouseX
      y=GraphicsWindow.MouseY
      setPoint()
      flag=0
    EndIf
  EndSub

    Sub testo
      GraphicsWindow.DrawText(10,50,"1) Clicca il tasto Start")
      GraphicsWindow.DrawText(10,70,"2) Clicca all'interno del
   triangolo")
    endsub

'procedura per disegnare ogni singolo pixel
Sub setPoint
    GraphicsWindow.SetPixel(x, y, 100)
    GraphicsWindow.SetPixel(x, y-1, 100)
    GraphicsWindow.SetPixel(x, y+1, 100)
    GraphicsWindow.SetPixel(x+1, y, 100)
    GraphicsWindow.SetPixel(x-1, y, 100)
  EndSub

  Sub repeat
    GraphicsWindow.Clear()
    GraphicsWindow.Height=1014
    GraphicsWindow.Width=1270
    GraphicsWindow.Top=0
    GraphicsWindow.Left=0
    GraphicsWindow.DrawTriangle(300, 800, 900,800,600,300)
    exit = Controls.AddButton("Chiudi",10,10)
    start = Controls.AddButton("Start",70,10)
    reset = Controls.AddButton("Ricomincia",120,10)
    flag=1
EndSub

'ciclo per estrarre un numero random
While 0=0
  c=math.GetRandomNumber(6)
  testo()
  If(flag=1) then
    If Controls.LastClickedButton = start then
      primoPunto()
    endif
  EndIf
  If(flag=0) then
    puntoMedio()
  EndIf
  If(Controls.LastClickedButton=exit) Then
    Program.End()
  EndIf
```

```
  If(Controls.LastClickedButton=reset) Then
    repeat()
  EndIf
EndWhile
```

Codice Triangolo 1/3

```
'disegna il triangolo
GraphicsWindow.Height=1014
GraphicsWindow.Width=1270
GraphicsWindow.Top=0
GraphicsWindow.Left=0
GraphicsWindow.DrawTriangle(600/2, 1100/2, 1200/2,1100/2,900/2,600/2)

exit = Controls.AddButton("Chiudi",10,10)
start = Controls.AddButton("Start",70,10)
reset = Controls.AddButton("Ricomincia",120,10)
flag=1

'procedura per trovare il punto medio
Sub puntoMedio
  If c=1 Or c=2 Then
    x=(Math.Abs(600+x)/3)
    y=(Math.Abs(1100+y)/3)
    setPoint()
  endif
  If c=3 Or c=4 Then
    x=(Math.Abs(1200+x)/3)
    y=(Math.Abs(1100+y)/3)
    setPoint()
  endif
  If c=5 Or c=6 Then
    x=(Math.Abs(900+x)/3)
    y=(Math.Abs(600+y)/3)
    setPoint()
  endif
EndSub

'procedura per disegnare il primo punto scelto con il mouse
Sub primoPunto
  If Mouse.IsLeftButtonDown Then
    x=GraphicsWindow.MouseX
    y=GraphicsWindow.MouseY
    setPoint()
    flag=0
  EndIf
EndSub
```

```
'procedura per disegnare ogni singolo pixel
Sub setPoint
    GraphicsWindow.SetPixel(x, y, 100)
    'GraphicsWindow.SetPixel(x, y-1, 100)
    'GraphicsWindow.SetPixel(x, y+1, 100)
    'GraphicsWindow.SetPixel(x+1, y, 100)
    'GraphicsWindow.SetPixel(x-1, y, 100)
  EndSub

  Sub repeat
    GraphicsWindow.Clear()
    GraphicsWindow.Height=1014
    GraphicsWindow.Width=1270
    GraphicsWindow.Top=0
    GraphicsWindow.Left=0
    GraphicsWindow.DrawTriangle(600/2, 1100/2,
   1200/2,1100/2,900/2,600/2)

    exit = Controls.AddButton("Chiudi",10,10)
    start = Controls.AddButton("Start",70,10)
    reset = Controls.AddButton("Ricomincia",120,10)
    flag=1
  EndSub

Sub testo
      GraphicsWindow.DrawText(10,50,"1) Clicca il tasto Start")
      GraphicsWindow.DrawText(10,70,"2) Clicca all'interno del
   triangolo")
    endsub

'ciclo per estrarre un numero random
While 0=0
  c=math.GetRandomNumber(6)
  testo()
  If(flag=1) then
    If Controls.LastClickedButton = start then
      primoPunto()
    endif
  EndIf
  If(flag=0) then
    puntoMedio()
  EndIf
  If(Controls.LastClickedButton=exit) Then
    Program.End()
  EndIf
  If(Controls.LastClickedButton=reset) Then
    repeat()
  EndIf
EndWhile
```

Codice Triangolo 1/4

```
'disegna il triangolo
GraphicsWindow.Height=1014
GraphicsWindow.Width=1270
GraphicsWindow.Top=0
GraphicsWindow.Left=0
GraphicsWindow.DrawTriangle(1300/3, 1800/3, 1900/3,1800/3, 1600/3,
   1300/3)

start = Controls.AddButton("Start",70,10)
reset = Controls.AddButton("Ricomincia",120,10)
exit = Controls.AddButton("Chiudi",10,10)

flag=1

'procedura per trovare il punto medio
Sub puntoMedio
  If c=1 Or c=2 Then
    x=(Math.Abs(1300+x)/4)
    y=(Math.Abs(1800+y)/4)
    setPoint()
  endif
  If c=3 Or c=4 Then
    x=(Math.Abs(1900+x)/4)
    y=(Math.Abs(1800+y)/4)
    setPoint()
  endif
  If c=5 Or c=6 Then
    x=(Math.Abs(1600+x)/4)
    y=(Math.Abs(1300+y)/4)
    setPoint()
  endif
EndSub

'procedura per disegnare il primo punto scelto con il mouse
Sub primoPunto
  If Mouse.IsLeftButtonDown Then
    x=GraphicsWindow.MouseX
    y=GraphicsWindow.MouseY
    setPoint()
    flag=0
  EndIf
EndSub

 Sub testo
      GraphicsWindow.DrawText(10,50,"1) Clicca il tasto Start")
```

```
      GraphicsWindow.DrawText(10,70,"2) Clicca all'interno del
   triangolo")
 endsub

'procedura per disegnare ogni singolo pixel
Sub setPoint
    GraphicsWindow.SetPixel(x, y, 100)
    'GraphicsWindow.SetPixel(x, y-1, 100)
    'GraphicsWindow.SetPixel(x, y+1, 100)
    'GraphicsWindow.SetPixel(x+1, y, 100)
    'GraphicsWindow.SetPixel(x-1, y, 100)
EndSub

 Sub repeat
    GraphicsWindow.Clear()
    GraphicsWindow.Height=1014
    GraphicsWindow.Width=1270
    GraphicsWindow.Top=0
    GraphicsWindow.Left=0
    GraphicsWindow.DrawTriangle(1300/3, 1800/3, 1900/3,1800/3, 1600/3,
   1300/3)
    exit = Controls.AddButton("Chiudi",10,10)
    start = Controls.AddButton("Start",70,10)
    reset = Controls.AddButton("Ricomincia",120,10)
    flag=1
EndSub

'ciclo per estrarre un numero random
While 0=0
  c=math.GetRandomNumber(6)
  testo()
  If(flag=1) then
    If Controls.LastClickedButton = start then
      primoPunto()
    endif
  EndIf
  If(flag=0) then
    puntoMedio()
  EndIf
  If(Controls.LastClickedButton=exit) Then
    Program.End()
  EndIf
  If(Controls.LastClickedButton=reset) Then
    repeat()
  EndIf
EndWhile
```

Codice Menù

```
public partial class MainWindow : Window
    {
        public MainWindow()
        {
            InitializeComponent();
            button_fratto_2.Visibility = Visibility.Hidden;
            button_fratto_3.Visibility = Visibility.Hidden;
            button_fratto_4.Visibility = Visibility.Hidden;
            label3.Visibility = Visibility.Hidden;
        }

        private void button1_Click(object sender, RoutedEventArgs e)
        {
            button_fratto_2.IsEnabled = true;
            button_fratto_3.IsEnabled = true;
            button_fratto_4.IsEnabled = true;
            button_fratto_2.Visibility = Visibility.Visible;
            button_fratto_3.Visibility = Visibility.Visible;
            button_fratto_4.Visibility = Visibility.Visible;
            label3.Visibility = Visibility.Visible;

        }
        private void pentagono_Click(object sender, RoutedEventArgs e)
        {
            System.Diagnostics.Process.Start("pentagono");
        }

        private void button_exit_Click(object sender, RoutedEventArgs
   e)
        {
            this.Close();
        }

        private void button_fratto_2_Click(object sender,
   RoutedEventArgs e)
        {

   System.Diagnostics.Process.Start("triangolo_di_sierpiski");
                button_fratto_2.IsEnabled = false;
                button_fratto_3.IsEnabled = false;
                button_fratto_4.IsEnabled = false;

        }

        private void button_fratto_3_Click(object sender,
   RoutedEventArgs e)
        {
```

```
        System.Diagnostics.Process.Start("triangolo_di_sierpiski_1divi
        so3");
                button_fratto_2.IsEnabled = false;
                button_fratto_3.IsEnabled = false;
                button_fratto_4.IsEnabled = false;

        }

        private void button_fratto_4_Click(object sender,
    RoutedEventArgs e)
        {

    System.Diagnostics.Process.Start("triangolo_di_sierpiski_1diviso4")
    ;
                button_fratto_2.IsEnabled = false;
                button_fratto_3.IsEnabled = false;
                button_fratto_4.IsEnabled = false;

        }

        private void albero_Click(object sender, RoutedEventArgs e)
        {
            System.Diagnostics.Process.Start("albero");
        }
     }
```

Codice H-Tree

```
GraphicsWindow.Width=Desktop.Width
GraphicsWindow.Height=Desktop.Height

Stack.PushValue("x1", 1000-400)
Stack.PushValue("y1", 700-200)
Stack.PushValue("alpha1", -Math.Pi/2)
Stack.PushValue("lung1", 150)

inclinazione=20
minLung=10

GraphicsWindow.PenColor="green"
GraphicsWindow.PenWidth=20
inizio=Controls.AddButton("Clicca per settare l'inclinazione (asse y)
   e la lunghezza minima dei rami (asse x)", 50, 50)

Sub Divisione2
  If(Stack.GetCount("x2")>0)Then
    angolo=Stack.PopValue("alpha2")
```

```
    lung=Stack.PopValue("lung2")/1.3
    GraphicsWindow.PenWidth=lung/10
    x=Stack.PopValue("x2")
    y=Stack.PopValue("y2")
    x1=x+lung*math.Cos(angolo+(Math.Pi/inclinazione))
    y1=y+lung*math.Sin(angolo+(Math.Pi/inclinazione))
    x2=x+lung*math.Cos(angolo-(Math.Pi/inclinazione))
    y2=y+lung*math.Sin(angolo-(Math.Pi/inclinazione))
    GraphicsWindow.DrawLine(x,y,x1,y1)
    GraphicsWindow.DrawLine(x,y,x2,y2)
    If(lung>minLung) Then
      Stack.PushValue("x1", x1)
      Stack.PushValue("x2", x2)
      Stack.PushValue("y1", y1)
      Stack.PushValue("y2", y2)
      Stack.PushValue("alpha1", angolo+(Math.Pi/inclinazione))
      Stack.PushValue("alpha2", angolo-(Math.Pi/inclinazione))
      Stack.PushValue("lung1", lung)
      Stack.PushValue("lung2", lung)
    EndIf
  EndIf
Endsub

Sub Divisione
  If(Stack.GetCount("x1")>0)Then
    angolo=Stack.PopValue("alpha1")
    lung=Stack.PopValue("lung1")/1.3
    GraphicsWindow.PenWidth=lung/10
    x=Stack.PopValue("x1")
    y=Stack.PopValue("y1")
    x1=x+lung*math.Cos(angolo+(Math.Pi/inclinazione))
    y1=y+lung*math.Sin(angolo+(Math.Pi/inclinazione))
    x2=x+lung*math.Cos(angolo-(Math.Pi/inclinazione))
    y2=y+lung*math.Sin(angolo-(Math.Pi/inclinazione))
    GraphicsWindow.DrawLine(x,y,x1,y1)
    GraphicsWindow.DrawLine(x,y,x2,y2)
    If(lung>minLung) Then
      Stack.PushValue("x1", x1)
      Stack.PushValue("x2", x2)
      Stack.PushValue("y1", y1)
      Stack.PushValue("y2", y2)
      Stack.PushValue("alpha1", angolo+(Math.Pi/inclinazione))
      Stack.PushValue("alpha2", angolo-(Math.Pi/inclinazione))
      Stack.PushValue("lung1", lung)
      Stack.PushValue("lung2", lung)
    EndIf
  EndIf
EndSub

Sub clear
```

```
  GraphicsWindow.Clear()
  While(Stack.GetCount("x1")>0)
    Stack.PopValue("x1")
  EndWhile
  While(Stack.GetCount("x2")>0)
    Stack.PopValue("x2")
  EndWhile
  While(Stack.GetCount("y1")>0)
    Stack.PopValue("y1")
  EndWhile
  While(Stack.GetCount("y2")>0)
    Stack.PopValue("y2")
  EndWhile
  While(Stack.GetCount("lung1")>0)
    Stack.PopValue("lung1")
  EndWhile
  While(Stack.GetCount("lung2")>0)
    Stack.PopValue("lung2")
  EndWhile
  While(Stack.GetCount("alpha1")>0)
    Stack.PopValue("alpha1")
  EndWhile
  While(Stack.GetCount("alpha2")>0)
    Stack.PopValue("alpha2")
  EndWhile
  minLung=mouse.MouseX
    If(minLung<50) then
      minLung=1
    Else
      minLung=minLung/50
    EndIf
    inclinazione=mouse.MouseY
    If(inclinazione<33) then
      inclinazione=1
    ElseIf inclinazione<100 then
      inclinazione=2
     Else
        inclinazione=inclinazione/33
      EndIf
    Stack.PushValue("x1", 1000-400)
    Stack.PushValue("y1", 700-200)
    Stack.PushValue("alpha1", -Math.Pi/2)
    Stack.PushValue("lung1", 150)
    GraphicsWindow.PenColor="green"
    GraphicsWindow.PenWidth=20
    GraphicsWindow.DrawLine(1000-400, 700-100, 1000-400, 700-200)
EndSub

Sub set
  For i=0 To GraphicsWindow.Width
```

```
    Controls.AddButton(Math.Round(i/50), i, 0)
    i=i+50
  EndFor
  For i=0 To GraphicsWindow.Height
    Controls.AddButton(Math.Round(i/33), 0, i)
    i=i+50
  EndFor
EndSub

While(0=0)
  If(Controls.LastClickedButton = inizio) Then
    clear()
    set()
    While(0=0)
      Divisione()
      Divisione2()
      If(Mouse.IsLeftButtonDown) Then
        clear()
        set()
     EndIf
    EndWhile
  EndIf
EndWhile
```

LIFE 1D

(linguaggio: Visual C#)

```
System.Windows.Forms.Label[] griglia = new Label[127]; // Array di Label...
const int N = 14;
const int NRIGHE = 8;
int c = 0;

//disattivare o attivare un bottone tramite true o false
Morti.Enabled = false;
Avanti.Enabled = false;

private void Form1_Load(object sender, EventArgs e)
{
          griglia[1] = label1;
       griglia[2] = label2;
   .
   .
   .
          griglia[125] = label125;
       griglia[126] = label126;
}


private void label1_Click(object sender, EventArgs e)
{
          label1.BackColor = Color.Black;
}
private void label2_Click(object sender, EventArgs e)
```

```
{

       label2.BackColor = Color.Black;
}
.
.
.
.
private void label125_Click(object sender, EventArgs e)
{
          label125.BackColor = Color.Black;
}
private void label126_Click(object sender, EventArgs e)
{
       label126.BackColor = Color.Black;
}



//Gestione bottone "chiudi"
private void button2_Click(object sender, EventArgs e)
{
          Close();

}

//Gestione bottone "clear”
private void Clear_Click(object sender, EventArgs e)
{
          label127.Text = "0";
          for (int i = 1; i <= 126; i++)
       {
          griglia[i].BackColor = Color.White;
       }
       c = 0;

       nascite.Enabled = true;
       Morti.Enabled = false;
       Avanti.Enabled = false;
}

//Gestione bottone "nascite"
private void nascite_Click(object sender, EventArgs e)
{
          for (int i = 1; i < 126; i++)
       {
          if (i == 1)
             {
                 if ((griglia[i + 1].BackColor == Color.Black) && (griglia[i].BackColor
                 != Color.Black))
                    {
                        griglia[i].BackColor = Color.Yellow;
                    }
             }
             if (i == 126)
                 {
              if ((griglia[i - 1].BackColor == Color.Black) && (griglia[i].BackColor !=
                 Color.Black))
                    {
                        griglia[i].BackColor = Color.Yellow;
                    }
              }
                if (i != 1 && i != 126)
                {
```

```
            if ((griglia[i + 1].BackColor == Color.Black || griglia[i - 1].BackColor ==
                Color.Black) && (griglia[i].BackColor != Color.Black))
                    {
                        griglia[i].BackColor = Color.Yellow;
                    }
                }

               }// Questa chiude il for..
            nascite.Enabled = false;
            Morti.Enabled = true;

        }

        //Gestione bottone "morti"
        private void Morti_Click(object sender, EventArgs e)
        {
                for (int i = 1; i < 126; i++)
                {
                    if (i == 1)
                    {
                     if ((griglia[i + 1].BackColor == Color.Black) && (griglia[i].BackColor !=
                        Color.Black))
                            {
                                griglia[i].BackColor = Color.Yellow;
                            }
                    }
                    if (i == 126)
                    {
                     if ((griglia[i - 1].BackColor == Color.White) && (griglia[i].BackColor ==
                        Color.Black))
                            {
                                griglia[i].BackColor = Color.White;

                            }
                    }
                    if (i != 1 && i != 126)
                    {
                     if ((griglia[i + 1].BackColor == Color.White && griglia[i - 1].BackColor ==
                        Color.White) && (griglia[i].BackColor == Color.Black))
                            {
                                griglia[i].BackColor = Color.AliceBlue;
                            }
                     if ((griglia[i + 1].BackColor == Color.Black && griglia[i - 1].BackColor ==
                        Color.Black) && (griglia[i].BackColor == Color.Black))
                            {
                                griglia[i].BackColor = Color.AliceBlue;
                            }
                     if ((griglia[i + 1].BackColor == Color.AliceBlue && griglia[i -
                        1].BackColor == Color.Black) && (griglia[i].BackColor == Color.Black))
                            {
                                griglia[i].BackColor = Color.AliceBlue;
                            }
                     if ((griglia[i + 1].BackColor == Color.Black && griglia[i - 1].BackColor ==
                        Color.AliceBlue) && (griglia[i].BackColor == Color.Black))
                            {
                                griglia[i].BackColor = Color.AliceBlue;

                            }
                }

               }// Questa chiude il for..
               Morti.Enabled = false;
            Avanti.Enabled = true;
```

```
 }

//Gestione bottone "avanti"
private void Avanti_Click(object sender, EventArgs e)
{
          for (int i = 1; i < 126; i++)
       {
          if (griglia[i].BackColor == Color.Yellow)
              {
                        griglia[i].BackColor = Color.Black;
              }
              if (griglia[i].BackColor == Color.AliceBlue)
              {
                    griglia[i].BackColor = Color.White;
              }
        }
        Morti.Enabled = false;
        Avanti.Enabled = false;
        nascite.Enabled = true;
        c++;
        label127.Text = Convert.ToString(c);
 }
```