GENERATING AND MODIFYING SCORES WITH GENERAL PURPOSE PROGRAMMING LANGUAGES

by Riccardo Bianchini

1. CHOOSING A PROGRAMMING LANGUAGE

If we want to generate or modify a Csound score by means of a general purpose programming language, we first have to choose a particular language to use. More robust and less robust languages exist; easy to learn languages, and languages that require a long time of study and practicing; languages which exist in all platforms, and languages specific to a single platform.

Interesting results can be achieved by using Artificial Intelligence languages (LISP, Prolog etc.), as these allow to define generating rules in a form that is close to human language.

We shall use, instead, BASIC, as it is easy to learn and is available on a large variety of computers and operating systems. The particular dialect we shall use is Microsoft Visual Basic.

2. WHAT SHELL WE NEED?

A programming language such as Visual Basic has a very large set of instructions and keywords. But we shall use only a little subset of these: the ones which deal with reading from and writing into a text file, the output format capability, some control structures (FOR... NEXT, WHILE... WEND and DO... LOOP WHILE), mathematical and logical expressions.

We shall launch Visual Basic, and in the Form/Load section we shall write the code. This event will occur when our application starts. As we shall not interact directly with the application, this is all we need.

3. LET’S WRITE A SCALE

Our first program simply generates a chromatic scale. Let’s see it:

```
Private Sub Form_Load()                 `this is provided by Visual Basic itself
Act     =   0                           `zeroing the variable which will contain action time
Dur     =   1                           `assigning the value 1 to duration
Amp     =   80                          `assigning the value 80 to amplitude
Open “score1.sco” For Output As #1      `open a file for writing, file id = 1
```

```
For i = 0 To 11                                   `For...Next loop beginning
        Pitch = 8 + i / 100                       `assign the pitch depending of the number of the note (i)
        Print #1, "i1 "; Act; " "; Dur; " "; Amp; " "; Pitch  `write into the file
        Act = Act + Dur                           `update the action time
Next i                                            `For...Next loop end
Close #1                                          `close the file
End Sub                                           `this is provided by Visual Basic itself
```

This little program will generate a score file named «score1.sco», the content of which will be:

```
i1      0       1       80      8
i1      1       1       80      8.01
i1      2       1       80      8.02
i1      3       1       80      8.03
i1      4       1       80      8.04
i1      5       1       80      8.05
i1      6       1       80      8.06
i1      7       1       80      8.07
i1      8       1       80      8.08
i1      9       1       80      8.09
i1      10      1       80      8.1
i1      11      1       80      8.11
```

Can we do better? Sure we can! Writing into the file, we separated each p-field with a blank space. Let's try separating the p-fields with tabs (in Basic, chr$(9)):

```
Private Sub Form_Load()                           `this is provided by Visual Basic itself
Act     =   0                                     `zeroing the variable which will contain action time
Dur     =   1                                     `assigning the value 1 to duration
Amp     =   80                                    `assigning the value 80 to amplitude
Open "score1.sco" For Output As #1                `open a file for writing, file id = 1
For i = 0 To 11                                   `For...Next loop beginning
        Pitch = 8 + i / 100                       `assign the pitch depending of the number of the note (i)
        Print #1, "i1 "; Act; chr$(9); Dur; chr$(9); Amp; chr$(9); Pitch    `write into the file
        Act = Act + Dur                           `update the action time
Next i                                            `For...Next loop end
Close #1                                          `close the file
End Sub                                           `this is provided by Visual Basic itself
```

Our score will be much better looking:

```
i1      0       1       80      8
i1      1       1       80      8.01
i1      2       1       80      8.02
i1      3       1       80      8.03
i1      4       1       80      8.04
i1      5       1       80      8.05
i1      6       1       80      8.06
i1      7       1       80      8.07
i1      8       1       80      8.08
i1      9       1       80      8.09
i1      10      1       80      8.01
i1      11      1       80      8.11
```

What if we want a chromatic scale with an accelerando? And what if the amplitude of each note decreases by 2 dB? Let’s rewrite our program for this:

```
Private Sub Form_Load()                                    `this is provided by Visual Basic itself
Act     =   0                                              `zeroing the variable which will contain action time
Dur     =   1                                              `assigning the value 1 to duration
Amp     =   80                                             `assigning the value 80 to amplitude
Open “score1.sco” For Output As #1                         `open a file for writing, file id = 1
For i = 0 To 11                                            `For...Next loop beginning
        Pitch = 8 + i / 100                                `assign the pitch depending of the number of the note (i)
        Print #1, “i1 “; Act; chr$(9); Dur; chr$(9); Amp; chr$(9); Pitch    `write into the file
        Act = Act + Dur                                    `update the action time
        Dur = Dur * 0.9                                    `for each new note, let’s diminish the duration
        Amp = Amp - 0.2                                    `for each new note, let’s diminish the amplitude
Next i                                                     `For...Next loop end
Close #1                                                   `close the file
End Sub                                                    `this is provided by Visual Basic itself
```

The generated score will be:

```
i1      0           1           80          8       i1      1       0.9     79.8    8.01
i1      1.9         0.81        79.6        8.02
i1      2.71        0.729       79.4        8.03
i1      3.439       0.6561      79.2        8.04
```

```
i1      4.0951          0.59049         79          8.05
i1      4.68559         0.531441        78.8        8.06
i1      5.217031        0.4782969       78.6        8.07
i1      5.6953279       0.43046721      78.4        8.08
i1      6.12579511      0.387420489     78.2        8.09
i1      6.513215599     0.3486784401    78          8.1
i1      6.8618940391    0.31381059609   77.8        8.11
```

What did we do? Simply, we did not tell Visual Basic to format the output properly, in such a way as to limit the number of decimal digits. Let's correct our program once more:

```
Private Sub Form_Load()                         `this is provided by Visual Basic itself
Act     =   0                                   `zeroing the variable which will contain action time
Dur     =   1                                   `assigning the value 1 to duration
Amp     =   80                                  `assigning the value 80 to amplitude
Open “score1.sco” For Output As #1              `open a file for writing, file id = 1
For i = 0 To 11                                 `For...Next loop beginning
        Pitch = 8 + i / 100                     `assign the pitch depending of the number of the note (i)
        Print #1, “i1 “; Format(Act,»##0.###»); chr$(9); Format(Dur,»###.###»); chr$(9); Format(Amp,»###.#»); chr$(9);
Format(Pitch,»###.##») `write into the file
        Act = Act + Dur                         `update the action time
        Dur = Dur * 0.9                         `for each new note, let's diminish the duration
        Amp = Amp - 0.2                         `for each new note, let's diminish the amplitude
Next i                                          `For...Next loop end
Close #1                                        `close the file
End Sub                                         `this is provided by Visual Basic itself
```

which will generate this score:

```
i1      0       1       80.     8.
i1      1       .9      79.8    8.01
i1      1.9     .81     79.6    8.02
i1      2.71    .729    79.4    8.03
i1      3.439   .656    79.2    8.04
i1      4.095   .59     79.     8.05
i1      4.686   .531    78.8    8.06
i1      5.217   .478    78.6    8.07
i1      5.695   .43     78.4    8.08
```

```
i1      6.126   .387    78.2    8.09
i1      6.513   .349    78.     8.1
i1      6.862   .314    77.8    8.11
```

4. LET’S COMPOSE A PIECE

What does *algorithmic composition* mean? Basically, it means the invention of a set of rules, and the composition of music which is generated by following these rules.

If rules are expressed in a language that the computer can «understand» (i.e. a *program*), the program can compose the piece by itself.

Let’s begin with a random piece. Visual Basic function *Rnd* generate a pseudo-random number in the range 0 ... 1. We shall use this function to determine the action time, the duration, the amplitude and the pitch of each note in our piece, limiting the random generation between some minimum and some maximum.

If in Visual Basic we write

```
x = Rnd
```

the variable x can assume any value between 0 and 1. Then we could write:

```
x = 8 + Rnd * 0.11
```

and the variable x can assume any value between 8 and 8.11, as 8 is a fixed part, to which a random number between 0 * 0.11 and 1 * 0.11 is added.

We can now write the program:

```
Private Sub Form_Load()                           `this is provided by Visual Basic itself
Open “c:\score1.sco” For Output As #1
Act     =    0                                    `zeroing the variable which will contain action time
For i = 1 To 20                                   `For...Next loop beginning
        Dur = 0.1 + Rnd * 0.4                     `generate a random duration between 0.1 and 0.5
        Amp = 60 + Rnd * 30                       `generate a random amplitude between 60 and 90
        Pitch = 8 + Rnd * 0.11                    `generate a random pitch between 8 and 8.11
        Print #1, “i1 “; Format(Act, “##0.###”); Chr$(9); Format(Dur, “###.###”); Chr$(9); Format(Amp, “###.#”); Chr$(9);
Format(Pitch, “###.##”) ‘write into the file
        Act = Act + Dur                         ‘update the action time
Next i
Close 1
End Sub
```

And we shall obtain the following score:

```
i1	0.	.382	76.	8.06
i1	0.382	.216	69.1	8.09
i1	0.598	.106	82.8	8.09
i1	0.704	.384	61.4	8.05
i1	1.087	.445	83.7	8.04
i1	1.532	.485	86.1	8.01
i1	2.017	.48	70.9	8.06
i1	2.497	.407	61.6	8.07
i1	2.904	.287	68.9	8.07
i1	3.191	.359	67.9	8.03
i1	3.55	.432	84.7	8.06
i1	3.982	.494	87.3	8.02
i1	4.477	.378	89.4	8.03
i1	4.855	.314	63.2	8.11
i1	5.168	.37	60.5	8.06
i1	5.539	.14	63.1	8.09
i1	5.679	.214	61.4	8.03
i1	5.893	.253	69.	8.1
i1	6.145	.492	72.	8.03
i1	6.637	.164	64.9	8.07
```

Let’s add a function table definition, e.g. a sine wave, and run Csound with a simple orchestra

```
sr	=	44100
kr	=	4410
ksmps	=	10
nchnls	=	1
	instr	1
ifrq	=	cpspch(p5)
iamp	=	ampdb(p4)
kenv	linseg	0, .01, iamp, p3-.01, 0
a1	oscil	kenv, ifrq, 1
	out	a1
	endin
```

and we can listen to a simple piece, in a style that could be defined «cagean».

A simple modification to the program will produce a polyphonic score. We can omit the line in which the program updates the action time, and have the computer generate a random action time:

```
Private Sub Form_Load()                          `this is provided by Visual Basic itself
Open “c:\score1.sco” For Output As #1
Act     =   0                                    `zeroing the variable which will contain action time
For i = 1 To 20                                  `For...Next loop beginning
      Act = Rnd * 20                             `generate a random action time between 0 and 20 secs
      Dur = 0.1 + Rnd * 0.4                      `generate a random duration between 0.1 and 0.5
      Amp = 60 + Rnd * 30                        `generate a random amplitude between 60 and 90
      Pitch = 8 + Rnd * 0.11                     `generate a random pitch between 8 and 8.11
      Print #1, “i1 “; Format(Act, “##0.###”); Chr$(9); Format(Dur, “###.###”); Chr$(9); Format(Amp, “###.#”); Chr$(9);
Format(Pitch, “###.##”) ‘write into the file
Next i
Close 1
End Sub
```

5. LET’S MODIFY AN EXISTING SCORE

A slightly more complex task is to modify an existing score, as we have to read it from a file, distinguish between the p-fields, modify it according to some rule, and finally write a new file.

If the required modification are local to each single note, (i.e. they do not depend on preceding or proceeding notes), we can read a note at a time, modify it and write it down into a new file. If this is not the case, we shall read the whole score, store all the p-fields in a multi-dimensional array, modify the notes and write them into the new file.

Suppose we want to alter all the amplitudes of a piece stored in the file ´score1.scoª, reducing them by 20%, and to write a new file called «score2.sco»;

```
Private Sub Form_Load()                          `this is provided by Visual Basic itself
Open “c:\score1.sco” For Input As #1             `open source file for reading, file id = 1
Open “c:\score2.sco” For Output As #2            `open destination file for writing, file id = 2
While not Eof(1)
      Input #1, a$                             `read «i1» and discard it
      Input #1, Act                              `read the action time
      Input #1, Dur                              `read the duration
      Input #1, Amp                              `read the amplitude
      Input #1, Pitch                            `read the pitch
```

```
        Amp = Amp * 0.8                                  `reduce the amplitude by 20%
Print #2, “i1 “; Format(Act, “##0.###”); Chr$(9); Format(Dur, “###.###”); Chr$(9); Format(Amp, “###.#”); Chr$(9);_
Format(Pitch, “###.##”) ‘write into the new file
Wend
Close 1
Close 2
End Sub
```

(Translated from the Italian by the author)