МАТЕМАТИКА И МАТЕМАТИЧЕСКО ОБРАЗОВАНИЕ, 2013
MATHEMATICS AND EDUCATION IN MATHEMATICS, 2013
Proceedings of the Forty Second Spring Conference
of the Union of Bulgarian Mathematicians
Borovetz, April 2–6, 2013

АЛГОРИТМИЧНА МУЗИКА В ЧАСОВЕТЕ ПО ИНФОРМАТИКА

Емил Келеведжиев, Зорница Дженкова

Предложен е учебен материал за средното училище, чрез който се свързват елементи от информатиката с музиката. Дадено е съкратено описание за използване на MIDI машина от програмисти на езика C и е направен увод в алгоритмичната музика. Подходът може да се прилага в извънкласната работа, а също и при бъдещо подобряване на задължителната учебна програма.

Увод. Осъвременяването на средното образование изисква използване на междупредметни връзки между отдалечени на пръв поглед области като информатика и музика. Както е известно, афинитетът на математиците към музиката е известен отдавна [1]. Различни изследвания показват, че децата, които свирят на пиано, проявяват и засилени логически способности, сходни с тези, които имат децата, отличаващи се с успешно решаване на сложни математически задачи. В последните години с навлизането на информационните технологии, почти всяко дете знае как да използва мобилния компютър, таблет или смартфон за слушане на музика или даже за обработката на музика чрез подходящ софтуер. Но въпреки практическата грамотност за музикалното използване на комютъра, за тези ученици, които изучават информатика, остават недостатъчно изяснени основните програмистки и алгоритмични постановки, чрез които софтуерът създава, обработва и възпроизвежда звуци и музика.

В предишна статия [2] сме разгледали построяването на музикалните гами и използване на интерфейса MIDI.

Средства и среди за възпроизвеждане на звукове. От многобройните възможности, ще представим две, които са подходяши за ученици, изучаващи алгоритмичен език.

1. *Средства при езика за програмиране C++ под операционната система Windows в пример на средата Dev-C++.* Бърз за усвояване начин се постига чрез мултимедийната библиотека на Майкрософт (Microsoft Win32 application programming interface (API)) за работа с интерфейса на MIDI машина (MIDI – Musical Instrument Digital Interface). Достъпът на ниско ниво се извършва с фунцията `midiOutShortMsg`, като е препоръчително да се подготвят интерфейсни функции `NoteOn, NoteOff, ProgramChange`. Описанието на параметрите е: `Channel` задава номер на канала за MIDI, който от своя страна определя тембъра на звучене. Това дава възможност за възпроизвеждане на различни музикални инструменти. Например, съгласно стандарта „General MIDI“ номер 1 съответства на традиционното пиано, 25 – на акустична китара, 74 – на флейта и т.н., `Pitch` задава височината на музикалния тон,

съгласно приета формула ([2]): `Pitch` $= 69 + 12\log_2(f/440)$ където f е честотата в Hz. Понеже 440 Hz е честотата на основния тон A („ла“), тоновете и полутоновете от основната октава съответстват на стойности на `Pitch` от 60 до 71, последователно за нотите: C, C#, D, D#, E, F, F#, G, G#, A, A# и B. `Velocity` определя „силата“ или „напрегнатостта“ на звучене, което може да бъде задавано различно при различните избори на `Channel`. Използваната средна стойност е 127.

```
#include <windows.h>
const int NOTE_ON = 0x90;
const int NOTE_OFF = 0x80;
const int PROGCHANGE = 0xC0;
const int CONTROLLER = 0xB0;
const int PITCHBEND = 0xE0;
HMIDIOUT hmo;
void NoteOn(int Channel, int Pitch, int Velocity)
{
 midiOutShortMsg (hmo,(DWORD)(Velocity * 0x10000) + (((Pitch*0x100) + (Channel-1)) + NOTE_ON));
}
void NoteOff(int Channel, int Pitch)
{
 midiOutShortMsg (hmo,(DWORD) (Pitch*0x100) + ((Channel-1) + NOTE_OFF));
}
void ProgramChange(int Channel, int Program)
{
  midiOutShortMsg (hmo, (DWORD) (Program * 0x100) + ((Channel-1) + PROGCHANGE ));
}
```

За улеснение може да подготвим фукция, с която да задаваме музикалния тон `p` и неговата продължителност `d`:

```
void play(int p, int d)
{ NoteOn(1,60+p,127); Sleep(d); NoteOff(1,60+p); }
```

след което да включим библиотеката `libwinmm.a` в проекта на Dev-C++ и да използваме следните инициализации в главната функция `main()`:

```
int main()
{   UINT mond = midiOutGetNumDevs(); // number of MIDI devices found
    if(mond==0) return 0;
    UINT_PTR      uDeviceID=0;
    LPMIDIOUTCAPS lpMidiOutCaps;
    MMRESULT mmr=midiOutGetDevCaps(uDeviceID,lpMidiOutCaps,sizeof(MIDIOUTCAPS));
//    cout << "Device chosen = " << lpMidiOutCaps->szPname << endl;
    mmr=midiOutOpen(&hmo, 0, 0, 0, CALLBACK_NULL);
    if(mmr !=0) return 0;
    play(0, 1000); // Възпроизвежда тон "до" в продължение на една секунда
    midiOutClose(hmo); system("pause");
}
```

Следният фрагмент показва как може да се програмира едновременно звучене на пиано и флейта:

```
    ProgramChange(1,1); ProgramChange(2,74);
    NoteOn(2,60,127); Sleep(1000);
    NoteOn(1,60,127); Sleep(1000);
    NoteOff(1,60);
    NoteOff(2,60);
```

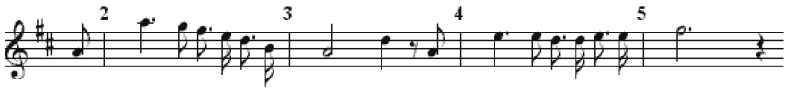
2. *Специализирана среда за програмиране на алгоритмична музика Vmm.* Този език е специално създаден за програмиране на музика и е близък по синтаксис с езика C. Наречен е Vmm (Virtual MIDI Machine)[3] и е свободно достъпен от цитирания сайт. Той е лесно усвоим за по-напредналите ученици, изучаващи програмиране.

Езикът и средата за работа с него предоставят средства за програмиране на алгоритми, с които да се управлява стандарния интерфейс MIDI.

Езикът Vmm е опростен в сравнение с повечето езици за общо предназначение. Той предлага само един тип променливи – числови, в които могат да се съхраняват както цели, така и дробни числа. На разположение на програмиста са масиви и функции, при което променливите могат да са локали и глобални. Функциите могат да се използват рекурсивно. Присъстват познатите от езика C аритметични и логически оператори, а от управляващите конструкции са застъпени `if`, `loop` и `while`.

Основно средство за управление са функциите `NoteOn()` и `NoteOff()`, за чието използване трябва да се включи библиотеката `stdMidi.vmm`. Параметрите в описанието на първата от тези функции: `NoteOn(<Channel>, <Pitch>, <Velocity>);` имат същия смисъл като описаните по-горе параметри на едноименната функция в средата за езика C++. Следният програмен фрагмент предизвиква звучене на тона C от основната октава в продължение на 2 секунди: `NoteOn(1,60,127); sleep(2000); NoteOff(1,60);` Цялостна програма на Vmm, която възпроизвежда началото на „Върви народе, възродени“, чийто нотен запис е даден на Фиг. 1, е следната:

```
#include "stdmidi.vmm"
proc main()
{ NoteOn(1,69,127); sleep(300); NoteOff(1,69);
  NoteOn(1,81,127); sleep(900); NoteOff(1,81);
  NoteOn(1,79,127); sleep(300); NoteOff(1,79);
  NoteOn(1,78,127); sleep(450); NoteOff(1,78);
  NoteOn(1,76,127); sleep(150); NoteOff(1,76);
  NoteOn(1,74,127); sleep(450); NoteOff(1,74);
  NoteOn(1,71,127); sleep(150); NoteOff(1,71);
  NoteOn(1,69,127); sleep(1200); NoteOff(1,69);
  NoteOn(1,74,127); sleep(600); NoteOff(1,74);
  sleep(300);
  NoteOn(1,69,127); sleep(300); NoteOff(1,69);
  NoteOn(1,76,127); sleep(900); NoteOff(1,76);
  NoteOn(1,76,127); sleep(300); NoteOff(1,76);
  NoteOn(1,74,127); sleep(450); NoteOff(1,74);
  NoteOn(1,74,127); sleep(150); NoteOff(1,74);
  NoteOn(1,76,127); sleep(450); NoteOff(1,76);
  NoteOn(1,76,127); sleep(150); NoteOff(1,76);
  NoteOn(1,78,127); sleep(1800); NoteOff(1,78);
}
```



Фиг. 1. Началото на химна „Върви народе, възродени“

Разбира се, от програмистка гледна точка много по-интересни са конструкции, в които по същество присъстват управляващи оператори. Предлагаме пример, в който със средствата на Vmm е програмирана „случайна музика“:

```
a; b; srand(GetSystemTime());
while(1)
{a = (rand()%20)+70; b = rand()%200; NoteOn(10,a,127); sleep(b); NoteOff(10,a);}
```

Рекурсията е едно от мощните средства в програмирането и е подходяща при алгоритмично създаване на музика. Предлагаме прост пример за рекурсивно генериране на числата на Фибоначи, където извикването на музикалните тонове става в рекурсивна функция:

```
proc fib(n)
 {if(n==0) {return (0)}; if(n==1) {return (1)};
  if(n>1)
   { p; p=fib(n-1)+fib(n-2);
     NoteOn(1,p%127,127); sleep(10*(p%10)); NoteOff(1,p%127);
     return (p); }
 }
proc main()
{ fib(100); }
```

Средата за програмиране Vmm поддържа квазипаралелни процеси, което е естествено необходимо при музикалните изпълнения – едновременно звучене на няколко музикални инструмента. Така могат да бъдат възпроизвеждани различни видове акорди, с които да се демонстрира консонантно или дисонантно звучене. Чрез звукови и музикални илюстрации може лесно и разбираемо за учениците да се въведе общата идея за паралелните процеси в информатиката.

Стартирането на два или повече квазипаралелни процеса в езика Vmm се извършва, като пред името на съответните функции написваме думата `thread`. Функциите, които искаме да се изпълняват едновременно, могат да бъдат различни или да бъде една и съща функция, извикана с различни параметри. В примера е показано как може да се възпроизведе едновременно хроматичната гама и нейно отместване с една октава.

```
proc p(b)
{i;i=0;
 while(i<12)
 { NoteOn(1,b+i,127); sleep(1000); NoteOff(1,b+i);
   i++; }
}
proc main()
{ thread p(60); thread p(72);}
```

Звукова илюстрация на класически алгоритми. Широко е известно и в преподаването често се прилага визуално илюстриране на протичането на процесите при изучаването на алгоритми. Място при „онагледяването“ има обаче и илюстрацията със звукови средства, самостоятелно или с комбинирането ѝ с визуализация. Звуковата илюстрация би намерила прием при учениците със засилено музикално възприемане, а също и при деца, страдащи от зрителни заболявания. Даваме няколко идеи:

1. *Пример при генериране на пермутации.* Функцията `next_permutation` е от стандартната библиотека STL. Ученикът може да чуе музикалния образ на всяка от пермутациите на числата от 1 до 4, както и да възприеме последователността им:

```
  const int n=4; int p[n];
  for(int i=0;i<n;i++) p[i] = i+1;
  do
  {for(int k=0;k<n;k++) { cout << p[k]; play(7*p[k],100);}
   cout << endl;
   Sleep(500);
  } while ( next_permutation (p,p+n) );
```

2. *Пример при изучаване на рекурсия.* Пресмятане на факториел с илюстриране на правия и обратен ход на рекурсията. Едновременно със звуците се изобразяват и числените стойности. Понеже факториелите растат много бързо, вместо стойностите им, се вземат техни логаритми.

```
int f(int n)
{ if(n==1) {system("pause"); return 1;}
  play(n,100);
  cout << n << endl;
  int p=n*f(n-1); play((int)log(p),100);
  cout << p << " " << (int)log(p) << endl;
  return p;
}
```

3. *Пример при обхождане на двоично дърво.* Илюстрира се построяване и обхождане на балансирано двоично дърво с възли, номерирани от 1 до 63 при извикване на `bt_CLR(1)`. Използваният метод CLR (center-left-right) може да бъде заменен например с LCR (left-center-right) и да се сравни музикалното звучене.

```
int c=1;
void bt_CLR(int lev)
{for(int i=0;i<=lev;i++) cout << " ";
 cout << c << endl; c++;
 play(2*lev,100);
 if(lev>5) return;
 bt_CLR(lev+1); bt_CLR(lev+1);
}
```

Алгоритмично композиране. В последните години все по-бързо се развиват и прилагат алгоритмични методи за композиране на музикални фрагменти и сигнали, а също и за цялостна музика, която да бъде изпълнявана и слушана. Заедно с това се засилват и съответните теоретични изследвания. В [4] е направен преглед на съвременните методи, които се използват за алгоритмично композиране. Те включват области на математиката и информатиката, свързани със стохастични процеси, вериги на Марков, хаотични системи, фрактали, клетъчни автомати, генетични алгоритми и др. От гледна точка на обучението в средното училище, достъпни за учениците са подходи, които не изискват задълбочена предварителна теория. Например методи, свързани с елементарна теория на числата, която се изучава в часовете по математика и подходи, отнасящи се за конструкции при изучаване на програмиране. Ще представим някои от тях, които основно са изградени върху алгоритмично генериране на числови редици:

1. *Пресмятане по формула в езика C.* В [5] и [6] са описани такива конструкции и са дадени примери, известни като „къси програми на C за генериране на музика“. Например, използват се стойности `t`, от които след пресмятане по формула с побитови операции се получава целочислена величина `p`, която след подходящо прилагане на модулна аритметика се подава като номер на тон за MIDI машината с помощта на интерфейсната фунция `play(p,d)`.

```
for(int t=1;;t++)
 {
  int p=
  (t*(t>>5|t>>8))>>(t>>16);
  p=-20+p%64; play(p,30);
 }
```

2. *Построение на Линденмайер (L-systems).* Конструкция [7], която има връзка с понятието „фракталност“, а също и с „формални граматики“, но може да бъде въведена и с елементартни средства. Като пример да разгледаме пораждащи правила от вида 1→(2,3); 2→(1,3); 3→(1,2), с помощта на които, започвайки от някоя от цифрите 1, 2 или 3, постепенно заместваме всяка цифра със съответната ѝ двойка: 1→23→1312→23122313→1312231313122312 и т.н, като спрем процеса до определе-

но ниво. След това от така получения низ, изпращаме всяка от цифрите му като стойност на тон в MIDI машина.

3. *Редица на Колац (Collatz)* ($3n + 1$ *проблем*). Редицата от числа, която се поражда от просто правило, описано в програмния фрагмент, е известна с хипотезата на Колац, че след краен брой итерации винаги ще се получи 1, независимо от началната стойност. За някои начални стойности (например $n = 27$) броят на итерациите може да бъде сравнително голям, което ще осигури по-дълго звучене.

```
ProgramChange(1,74); int n=27;
while(n>1)
{int n1;
 if(n%2==0) n1=n/2; else n1=3*n+1;
 cout << n << " ";
 play(n%16,100*(n/n1));
 n=n1;
}
```

4. *Редицата от простите числа.* Учениците са запознати с числата (наречени *прости*), които нямат делители, различни от едно и от себе си. Интересно е, че тези числа може да бъдат в основата на програмиране на музика.

```
for(int t=1;;t++)
 if (is_prime(t))
  { cout << t << endl;
    int p=t; p=-20+p%64;
    play(p,10);
  }
```

Софтуерни средства за създаване на алгоритмична музика. Освен разгледаните по-горе езици за програмиране, съществуват и други, подходящи за алгоритмично композиране. При универсалните езици употребата им за алгоритмична музика зависи от налични за това библиотеки. Анотиран списък на *специализирани езици* за музикално композиране е даден в [8]. Измежду тях ще споменем езика за музикално композиране и звуков дизайн Csound. Той позволява лесна имплементация на различни алгоритмични процеси и е добро средство за експериментиране със звукови параметри. Друга интересна програмна среда за създаване на музика е Max/MSP. С нейният обектно-ориентиран език лесно се създават почти безкрайни варианти на приложения. Да отбележим, че за ученици от началните класове все по-голяма популярност за програмиране на музика придобива езикът Scratch (виж българсия сайт [9]).

Понастоящем по света са разработени и разпространени многобройни компютърни приложения с модерен и дружелюбен интерфейс за създаване и експериментиране с алгоритмична музика – с тях не е задължително потребителят да притежава умения за работа с език за програмиране. Налични са програми, както за PC платформи с операционни системи Windows и Mac, така и за таблети и смартфони с Android, iOS и др. Една част от този софтуер е платена, но има и достатъчно добри безплатни продукти и всички те биха заинтересували учители и ученици. В работата [10] може да бъде намерен добър аналитичен обзор за голям брой такива софтуерни продукти за PC платформи, например FractMus, Fractal Composer, Art Song, MusicLab, MusiNum и др. От продуктите за таблети и смартфони ще спомена BitWiz, Audio Synth и MixTikl. Разработени са и уеб базирани генератори на музика, например WolframTones.

Поради ограничения обем на настоящето изложение не е възможно да дадем подробностите за всички споменати среди и езици, но се надяваме, че интересуващият се читател ще бъде подпомогнат от цитираните препратки.

Заключение. С елементи от изложения материал сме запознали участници в Ученическия институт по математика и информатика, и той е послужил за основа на ученически разработки. Надяваме се да намерим последователи сред учителите у нас за бързо развиващото се в света направление на алгоритмичната музика. Да

доближим към музиката тези ученици, които имат повишен интерес към математиката и програмирането, както и обратното – да спечелим нови привърженици на информатиката измежду учениците и преподавателите, които се увличат в музикалното изкуство.

ЛИТЕРАТУРА

[1] E. Bloch. Essays on the Philosophy of Music. Cambridge, UK, Cambridge University Press, 1985.

[2] Е. Келеведжиев, З. Дженкова. Математетика, информатика и музика. *Математика и математическо образование*, **36** (2007), 92–101.

[3] Virtual Midi Machine (accessed on 10 Nov 2012). http://vmm.audionetwork.be/

[4] G. Diaz-Jerez. Algorithmic Music: Using Mathematical Models in Music Composition. Doctor Thesis. Manhattan School of Music, August 2000 (accessed on 11 Nov 2012). http://www.gustavodiazjerez.com/Gustavo_Diaz_Jerez_DMA_Thesis_MSM.pdf

[5] Ville-Matias Heikkilëa. Discovering novel computer music techniques by exploring the space of short computer programs (accessed on 11 Nov 2012). (arXiv:1112.1368v1[cs.SD]) http://arxiv.org/abs/1112.1368

[6] Ville-Matias Heikkilëa. Collection of oneliner music formulas (accessed on 11 Nov 2012). http://pelulamu.net/countercomplex/music_formula_collection.txt

[7] G. Rozenberg, A. Salomaa. L-systems. Encyclopedia of Mathematics. (accessed on 11 Nov 2012) http://www.encyclopediaofmath.org/index.php?title=L-systems

[8] Programming Languages Used for Music (accessed on 8 Jan 2013) http://www.nosuch.com/tjt/plum.html

[9] Сайт за програмния език Scratch за деца, учители и родители. (accessed on 8 Jan 2013) http://www.scratch.bg

[10] Стела Атанасова. Изследване на методи и средства за създаване на компютърна алгоритмична музика. Дисератация. (accessed on 8 Jan 2013) http://eprints.nbu.bg/1379/1/DISERTAZIA%20STELA%20NEW.pdf

Емил Келеведжиев
Институт по математика и информатика
Българска академия на науките
ул. Акад. Г. Бончев, бл. 8
1113 София
e-mail: keleved@math.bas.bg

Зорница Дженкова
6 ОУ „Иван Вазов“
ул. Митко Палаузов № 54
5300 Габрово
e-mail: zornitsa.dzhenkova@gmail.com

ALGORITHMIC MUSIC WHEN TEACHING INFORMATICS

Emil Kelevedjiev, Zornitsa Dzhenkova

Elements of secondary school material are offered for unified teaching in some aspects of computer programming and music. A short description of a technique oriented to the MIDI machine for C programmers is given and an introduction to algorithmic music is outlined. The approach could be applied in extra-curriculum activities as well as a proposal for a future improvement in the compulsory curriculum.