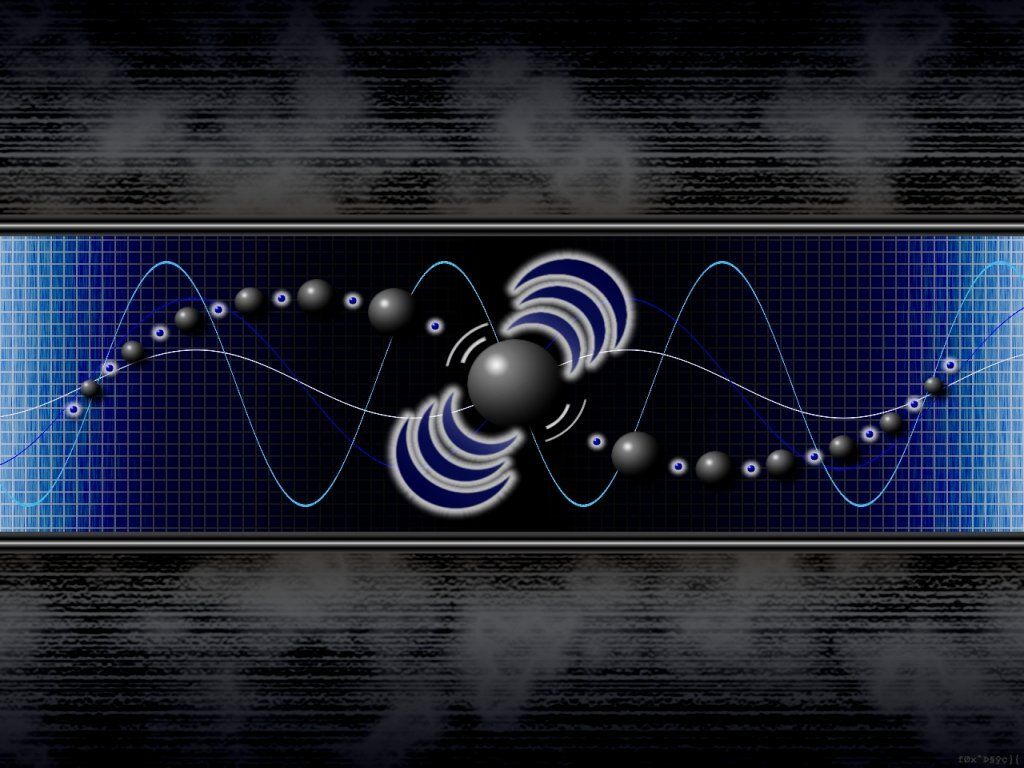


ПРАВИТЕЛЬСТВО САНКТ-ПЕТЕРБУРГА

КОМИТЕТ ПО ОБРАЗОВАНИЮ

Как звучат функции

Руководитель проекта

Андреева Ольга Алексеевна

Работу выполнила

Абрамова Ирина,

ученица 8-в класса.

г. Санкт-Петербург

2014 г

Содержание

1. Введение
2. Объект и предмет исследования
3. Цифровой звук
4. Программирование звука
5. Генерация мелодии
6. Стандартные функции
7. Выводы
8. Источники информации
9. Приложения
10. **Введение**

Функции — это математические портреты устойчивых закономерностей, познаваемых человеком. Чтобы проиллюстрировать характерные свойства функций, нам показалось интересным их озвучить.

Интерес к программированию привел меня к замыслу: использовать возможности языка программирования Pascal для озвучивания стандартных (всем известных) функций. В ходе работы стало очевидно, что многие звуковые сочетания в окружающем нас мире напоминают «звучание» таких функций

Цель работы: использовать возможности языка программирования   
Pascal для озвучивания стандартных математических функций.

Задачи:

* узнать больше о природе звука;
* освоить процедуры языка Pascal для создания звукоряда;
* написать программу для цифровых нот;
* получить цифровое звучание простых мелодий;
* написать программу для озвучивания стандартных математических функций;
* обработать полученные материалы и подготовить отчет.

1. **Объект и предмет исследования**

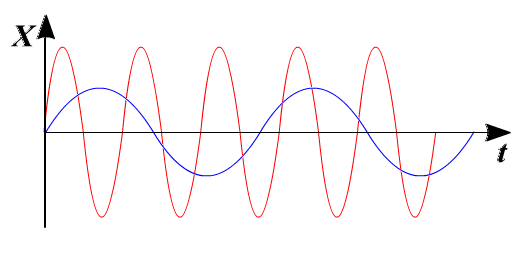
Объектом исследования является цифровой звук и возможности языка программирования по управлению цифровым звуком.

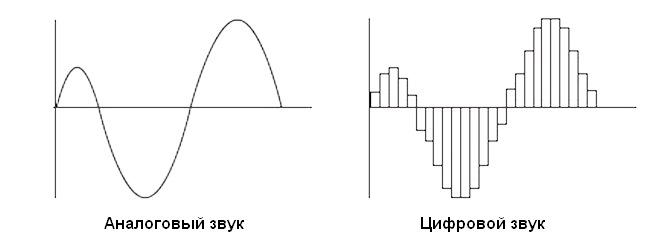
Предмет исследования – стандартные математические функции как основа для получения звукоряда.

1. **Цифровой звук**

Звук – это механические колебания упругой среды, воспринимаемые органами слуха. Главные характеристики звуковой волны:

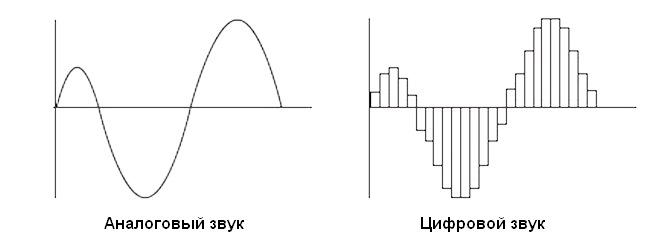
1. амплитуда сигнала – она определяет силу звука;

2 частота колебаний – она определяет высоту звука.



**X**

**t**



Если эти колебания воспроизводятся техническими приборами под управлением компьютера, то мы получаем цифровой звук.

В основе цифрового звука лежат двоичные коды, задающие основные параметры будущего звука.

Получить эти коды можно путем обработки – оцифровки – природных звуков, таких как голос, звучания музыкального инструмента, звуки природы. Другой способ ⎯ создать их с помощью программирования, что и является содержанием данной работы.

В результате этого процесса появляется возможность компьютерной обработки звуковых сигналов и получения звуковых композиций.

1. **Программирование звука**

Для генерации звуков в среде программирования Pascal имеются следующие процедуры:

**Sound (X)** – воспроизводит непрерывный звук частотой X Гц;

**NoSound** – отключает звук;

**Delay (X)** – задает длительность звучания в миллисекундах.

Для построения звукоряда мы использовали диапазон первой и второй октавы: от 200 Гц до 1100 Гц. Эти звуки наиболее приятны для уха человека. В таблице приведены частоты звуков первой и второй октавы.

|  |  |  |
| --- | --- | --- |
| **Нота** | **Частота (Гц)** | |
| 1 октава | 2 октава |
| До | 262 | 524 |
| Ре | 294 | 598 |
| Ми | 330 | 660 |
| ФА | 350 | 770 |
| Соль | 392 | 794 |
| Ля | 440 | 880 |
| Си | 494 | 898 |

Однако для эффективного управления звуком понадобилось разработать свои процедуры.

Процедура MS( hz, t) задаёт:

hz - частоту звука в герцах

t - длительность звучания ноты.

Кроме того, понадобились паузы между звуками, которые задает другая процедура PAUSE(t) – длительность паузы также задается в долях такта.

**procedure pause (t :byte);**

**begin**

**delay (round (1000 / t));**

**end;**

**procedure MS ( hz: integer; t: real );**

**var ti: real;**

**begin**

**if t>0 then**

**begin ti := 1000 / t;**

**Sound ( hz ); Delay ( round( ti ));**

**NoSound;**

**end;**

**end;**

В Приложении 1 приведена программа генерации цифровых нот  
первой октавы.

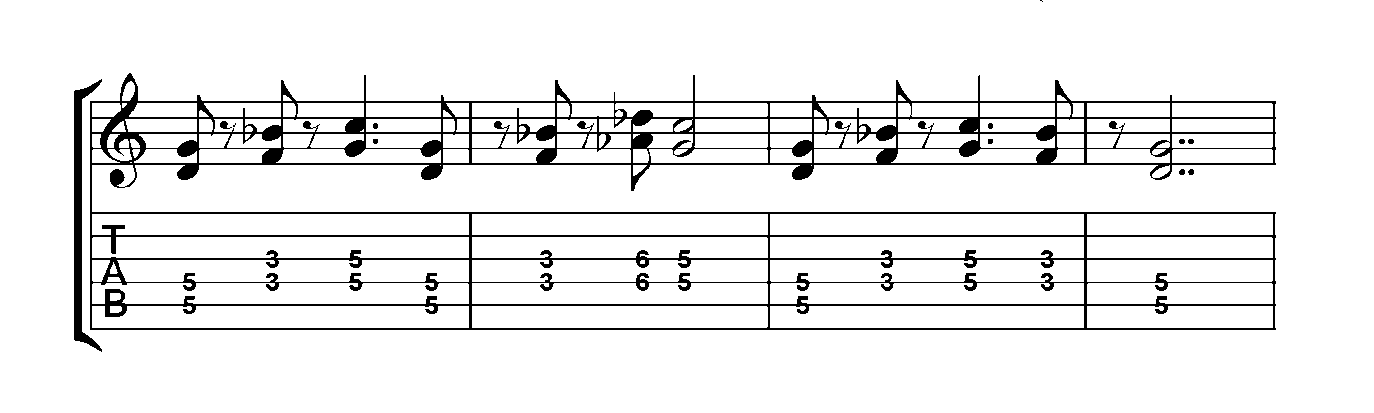
1. **Генерация мелодии**

В качестве эксперимента были запрограммированы фрагменты несложных мелодий: каждой ноте из нотной записи мелодии соответствует цифровой звук с нужными параметрами.

Первый фрагмент из песни «Пусть всегда будет солнце»,

второй – из песни «Smoke on the water».



**«Smoke on the water»**

В Приложении 2 и Приложении 3 приведены программы генерации этих музыкальных фрагментов.

1. **Стандартные функции**

Для озвучивания взяты стандартные функции, которые легко реализовать средствами Pascal.

|  |  |
| --- | --- |
| Название | Функция |
| Линейная |  |
| Кусочно-линейная |  |
| Квадратичная |  |
| Гармоническая |  |
| Случайная |  |

Разработанная программа на языке Pascal приведена в Приложении 4. Для прослушивания звукоряда необходимы звуковые колонки (наушники).

1. **Выводы**

В результате проделанной работы были сделаны следующие выводы:

* возможностей языка программирования достаточно для получения звуков и простых мелодий, которые сопровождают работу, например гаджетов;
* рассмотренные функциональные зависимости находят широкое отражение в жизни - полученный звукоряд позволяет распознать их на слух;
* кроме того, тема исследования позволила проследить взаимосвязь знаний из разных областей деятельности человека;
* мною приобретены новые знания в области физики, музыки, математики, информатики;
* возрос интерес к самостоятельной исследовательской работе;

1. **Источники информации**
2. Комлев Н.Ю.«Самоучитель игры на Паскаль. АВС и немного Турбо»-М: Солон-Пресс, 2013.
3. http://www.7not.ru/theory/01.phtml
4. http://www.bibliofond.ru/view.aspx?id=7236

**Приложение 1. Цифровые ноты**

**Program Noty;**

**uses crt; var i : byte;**

**procedure MS( hz : integer; t : real );**

**var ti: real;**

**begin**

**if t > 0 then**

**begin ti := 2000 / t;**

**Sound( hz ); Delay( round( ti)); NoSound;**

**end;**

**end;**

**begin**

**for i := 0 to 88 do MS( 50 \* i, 24) end;**

**MS( 262, 2); MS( 294, 2); MS( 330, 2); MS( 349,2);**

**MS( 392, 2); MS( 440, 2); MS( 494, 2);**

**end.**

**Приложение 2. Генерация мелодий**

**Program Pust\_vsegda\_budet;**

**uses crt;**

**procedure pause( t: byte);**

**begin**

**delay( round( 1000 / t));**

**end;**

**procedure MS(hz: integer; t: real);**

**var ti:real;**

**begin**

**if t>0 then**

**begin**

**ti := 2000 / t; Sound( hz ); Delay(round( ti )); NoSound;**

**end;**

**end;**

**begin**

**MS(294, 4); pause(16); MS(294,4);**

**MS(392, 2); pause(24); MS(440,4);MS(494,4);**

**MS(440, 4); MS(392, 4); pause(24); MS(294,4); pause(16);**

**MS(294, 4); MS(392, 2); pause(24); MS(440,4);MS(494,4);**

**MS(494, 4); MS(440, 4); pause(24); MS(294,4); pause(16); MS(294, 4);**

**MS(440, 2); pause(24); MS(494, 4);MS(522, 4);**

**MS(600, 4); MS(440, 4); pause(24); MS(440, 4); pause(16); MS(494, 4);**

**MS(522, 2); pause(24); MS(494, 4);MS(440, 4);**

**MS(392, 2);**

**end.**

**Приложение 3. Генерация мелодий**

**Program Smoke\_on\_the\_water;**

**uses crt;**

**procedure MS( hz : integer; t : real);**

**var ti: real;**

**begin**

**if t>0 then**

**begin ti := 2000 / t;**

**Sound( hz ); Delay(round( ti ) ); NoSound;**

**end;**

**end;**

**procedure pause( t: byte);**

**begin**

**delay(round(2000 / t));**

**end;**

**begin**

**MS(330, 8); pause(8); MS(392, 8); pause(8); MS(440, 8/3); pause(16);**

**MS(330, 8); pause(8); MS(393, 8); pause(8); MS(465, 8); MS(440, 8/3); pause(8);**

**MS(330, 8); pause(8); MS(392, 8); pause(8); MS(440, 8 / 3);pause(16);**

**MS(392, 8); pause(8); MS(330, 8 / 3); pause(8);**

**end.**

**Приложение 4. Программа озвучивания функций**

**Program Kak\_zvuchat\_funcci;**

**uses crt;**

**var t : integer; j, i : integer; s : real;**

**procedure pause(t: byte);**

**begin**

**delay(round(1000 / t));**

**end;**

**procedure MS(hz : integer; t : real);**

**var ti : real;**

**begin**

**if t > 0 then**

**begin ti := 2000 / t;**

**Sound( hz); Delay(round( ti)); NoSound;**

**end;**

**end;**

**procedure Ran\_Dom;**

**begin**

**for i := 1 to 20 do**

**begin**

**t := random(5) \* 10 + 150; MS(t, 8); pause(16);**

**end;**

**end;**

**procedure PILA\_N;**

**begin**

**for i := 1 to 8 do**

**begin**

**t := (i mod 2) \* 400 + 200; MS(t, 4); pause(8);**

**end;**

**end;**

**procedure PILA\_V;**

**begin**

**for j := 1 to 4 do**

**begin**

**for i := 9 downto 1 do**

**begin**

**t := round( I \* 60 + 150); MS(t, 16);**

**end;**

**for I := 1 to 9 do**

**begin**

**t := round( I \* 60 + 150); MS(t, 16);**

**end;**

**end;**

**for i:=9 DOWNto -1 do**

**begin**

**t:=round(I \* 60 + 150); MS(t, 16);**

**end;**

**end;**

**procedure PARABOLA;**

**begin**

**for I := -30 to 31 do**

**begin**

**t := I \* I + 70; MS(t, 32);**

**end;**

**pause( 1);**

**for I := 30 downto -30 do**

**begin t := 970 – I \* I ; MS(t, 32); end;**

**pause(1);**

**end;**

**procedure SIN\_3;**

**begin**

**for j := 1 to 3 do**

**for I := 0 to 100 do**

**begin**

**t := round(sin( I \* 6.28 / 100) \* 200) + 400; MS(t, 96)**

**end;**

**pause(2);**

**end;**

**begin**

**randomize;**

**ClrScr;**

**PILA\_N; readln;**

**PILA\_V; readln;**

**PARABOLA; readln;**

**SIN\_3; readln;**

**Ran\_dom; readln;**

**end.**