```
int buttonState2= 0;

const int buttonPin2=5;

 const int buzzer= 11;

void setup() {

 #define NOTE_B0  31
#define NOTE_C1  33
#define NOTE_CS1 35
#define NOTE_D1  37
#define NOTE_DS1 39
#define NOTE_E1  41
#define NOTE_F1  44
#define NOTE_FS1 46
#define NOTE_G1  49
#define NOTE_GS1 52
#define NOTE_A1  55
#define NOTE_AS1 58
#define NOTE_B1  62
#define NOTE_C2  65
#define NOTE_CS2 69
#define NOTE_D2  73
#define NOTE_DS2 78
#define NOTE_E2  82
#define NOTE_F2  87
#define NOTE_FS2 93
#define NOTE_G2  98
#define NOTE_GS2 104
#define NOTE_A2  110
#define NOTE_AS2 117
#define NOTE_B2  123
#define NOTE_C3  131
#define NOTE_CS3 139
#define NOTE_D3  147
#define NOTE_DS3 156
#define NOTE_E3  165
#define NOTE_F3  175
#define NOTE_FS3 185
#define NOTE_G3  196
#define NOTE_GS3 208
#define NOTE_A3  220
```

```
#define NOTE_AS3 233
#define NOTE_B3  247
#define NOTE_C4  262
#define NOTE_CS4 277
#define NOTE_D4  294
#define NOTE_DS4 311
#define NOTE_E4  330
#define NOTE_F4  349
#define NOTE_FS4 370
#define NOTE_G4  392
#define NOTE_GS4 415
#define NOTE_A4  440
#define NOTE_AS4 466
#define NOTE_B4  494
#define NOTE_C5  523
#define NOTE_CS5 554
#define NOTE_D5  587
#define NOTE_DS5 622
#define NOTE_E5  659
#define NOTE_F5  698
#define NOTE_FS5 740
#define NOTE_G5  784
#define NOTE_GS5 831
#define NOTE_A5  880
#define NOTE_AS5 932
#define NOTE_B5  988
#define NOTE_C6  1047
#define NOTE_CS6 1109
#define NOTE_D6  1175
#define NOTE_DS6 1245
#define NOTE_E6  1319
#define NOTE_F6  1397
#define NOTE_FS6 1480
#define NOTE_G6  1568
#define NOTE_GS6 1661
#define NOTE_A6  1760
#define NOTE_AS6 1865
#define NOTE_B6  1976
#define NOTE_C7  2093
#define NOTE_CS7 2217
#define NOTE_D7  2349
#define NOTE_DS7 2489
#define NOTE_E7  2637
#define NOTE_F7  2794
```

```
#define NOTE_FS7 2960
#define NOTE_G7  3136
#define NOTE_GS7 3322
#define NOTE_A7  3520
#define NOTE_AS7 3729
#define NOTE_B7  3951
#define NOTE_C8  4186
#define NOTE_CS8 4435
#define NOTE_D8  4699
#define NOTE_DS8 4978
#define REST 0



pinMode( buttonPin2, INPUT_PULLUP);

pinMode(buzzer, OUTPUT);

}
void loop()
{

  buttonState2=digitalRead(buttonPin2);
randomSeed(analogRead(A0));
int num=rand()%15;


if(buttonState2==LOW)
{

Serial.begin(9600);

Serial.println(num);
int divider = 0, noteDuration = 0;
int tempo=144;
 int wholenote = (60000 * 4) / tempo;

if(num==0)
{


int melody[] = {
NOTE_E5, 4,  NOTE_B4,8,  NOTE_C5,8,  NOTE_D5,4,  NOTE_C5,8,  NOTE_B4,8,
  NOTE_A4, 4,  NOTE_A4,8,  NOTE_C5,8,  NOTE_E5,4,  NOTE_D5,8,  NOTE_C5,8,
```

```
  NOTE_B4, -4,  NOTE_C5,8,  NOTE_D5,4,  NOTE_E5,4,
  NOTE_C5, 4,  NOTE_A4,4,  NOTE_A4,8,  NOTE_A4,4,  NOTE_B4,8,  NOTE_C5,8,

  NOTE_D5, -4,  NOTE_F5,8,  NOTE_A5,4,  NOTE_G5,8,  NOTE_F5,8,
  NOTE_E5, -4,  NOTE_C5,8,  NOTE_E5,4,  NOTE_D5,8,  NOTE_C5,8,
};




int notes = sizeof(melody) / sizeof(melody[0]) / 2;
int divider = 0, noteDuration = 0;
for (int thisNote = 0; thisNote < notes * 2; thisNote = thisNote + 2)
 {
divider = melody[thisNote + 1];
    if (divider > 0) {
 noteDuration = (wholenote) / divider;
}
  else if (divider < 0)
  {
    noteDuration = (wholenote) / abs(divider);
      noteDuration *= 1.5;
  }
   tone(buzzer, melody[thisNote], noteDuration*0.9);
 delay(noteDuration);
   noTone(buzzer);
}



}
else if(num==2)
{
  int melody[]={
    NOTE_FS5,8, NOTE_FS5,8,NOTE_D5,8, NOTE_B4,8, REST,8, NOTE_B4,8, REST,8,
NOTE_E5,8,
  REST,8, NOTE_E5,8, REST,8, NOTE_E5,8, NOTE_GS5,8, NOTE_GS5,8, NOTE_A5,8,
NOTE_B5,8,
  NOTE_A5,8, NOTE_A5,8, NOTE_A5,8, NOTE_E5,8, REST,8, NOTE_D5,8, REST,8,
NOTE_FS5,8,

  };
  int notes = sizeof(melody) / sizeof(melody[0]) / 2;
```

```
for (int thisNote = 0; thisNote < notes * 2; thisNote = thisNote + 2)
 {
divider = melody[thisNote + 1];
    if (divider > 0) {
 noteDuration = (wholenote) / divider;
}
  else if (divider < 0)
  {
    noteDuration = (wholenote) / abs(divider);
      noteDuration *= 1.5;
  }

    tone(buzzer, melody[thisNote], noteDuration*0.9);
 delay(noteDuration);
   noTone(buzzer);
}
}
else if(num==1)
{
   int tempo = 85;
int melody[] =
{


  NOTE_G4,8, NOTE_C4,8, NOTE_E4,16, NOTE_F4,16, NOTE_G4,8, NOTE_C4,8,
NOTE_E4,16, NOTE_F4,16,
  NOTE_G4,8, NOTE_C4,8, NOTE_E4,16, NOTE_F4,16, NOTE_G4,8, NOTE_C4,8,
NOTE_E4,16, NOTE_F4,16,
  NOTE_G4,-4, NOTE_C4,-4,//5
  NOTE_DS4,16, NOTE_F4,16, NOTE_G4,4, NOTE_C4,4, NOTE_DS4,16, NOTE_F4,16, //6
  NOTE_D4,-1, //7 and 8




};

 int wholenote = (60000 * 4) / tempo;

int notes = sizeof(melody) / sizeof(melody[0]) / 2;
```

```
for (int thisNote = 0; thisNote < notes * 2; thisNote = thisNote + 2)
 {
divider = melody[thisNote + 1];
    if (divider > 0) {
 noteDuration = (wholenote) / divider;
}
  else if (divider < 0)
  {
    noteDuration = (wholenote) / abs(divider);
      noteDuration *= 1.5;
  }
   tone(buzzer, melody[thisNote], noteDuration*0.9);
 delay(noteDuration);
   noTone(buzzer);
}
}
else if(num==3)
{

int melody[] = {
  REST, 2, NOTE_D4, 4,
  NOTE_G4, -4, NOTE_AS4, 8, NOTE_A4, 4,
  NOTE_G4, 2, NOTE_D5, 4,
  NOTE_C5, -2,
  NOTE_A4, -2,
  NOTE_G4, -4, NOTE_AS4, 8, NOTE_A4, 4,
  NOTE_F4, 2, NOTE_GS4, 4,
  NOTE_D4, -1,
  NOTE_D4, 4,

};

int notes = sizeof(melody) / sizeof(melody[0]) / 2;

for (int thisNote = 0; thisNote < notes * 2; thisNote = thisNote + 2)
 {
divider = melody[thisNote + 1];
    if (divider > 0) {
 noteDuration = (wholenote) / divider;
}
  else if (divider < 0)
  {
    noteDuration = (wholenote) / abs(divider);
      noteDuration *= 1.5;
```

```
  }
   tone(buzzer, melody[thisNote], noteDuration*0.9);
 delay(noteDuration);
   noTone(buzzer);
}
}
else if (num==4)
{
  int melody[] = {NOTE_FS4,8, REST,8, NOTE_A4,8, NOTE_CS5,8, REST,8,NOTE_A4,8,
REST,8, NOTE_FS4,8, //1
  NOTE_D4,8, NOTE_D4,8, NOTE_D4,8, REST,8, REST,4, REST,8, NOTE_CS4,8,
  NOTE_D4,8, NOTE_FS4,8, NOTE_A4,8, NOTE_CS5,8, REST,8, NOTE_A4,8, REST,8,
NOTE_F4,8,
  NOTE_E5,-4, NOTE_DS5,8, NOTE_D5,8, REST,8, REST,4,};

  int tempo=114;
  int notes = sizeof(melody) / sizeof(melody[0]) / 2;

for (int thisNote = 0; thisNote < notes * 2; thisNote = thisNote + 2)
 {
divider = melody[thisNote + 1];
    if (divider > 0) {
 noteDuration = (wholenote) / divider;
}
  else if (divider < 0)
  {
    noteDuration = (wholenote) / abs(divider);
      noteDuration *= 1.5;
  }
   tone(buzzer, melody[thisNote], noteDuration*0.9);
 delay(noteDuration);
   noTone(buzzer);
}
}
else if (num==5)
{
  int melody[] = {
  NOTE_A4,16, NOTE_B4,16, NOTE_D5,16, NOTE_B4,16,
  NOTE_FS5,-8, NOTE_FS5,-8, NOTE_E5,-4, NOTE_A4,16, NOTE_B4,16, NOTE_D5,16,
NOTE_B4,16,
   NOTE_E5,-8, NOTE_E5,-8, NOTE_D5,-8, NOTE_CS5,16, NOTE_B4,-8, NOTE_A4,16,
NOTE_B4,16, NOTE_D5,16, NOTE_B4,16, //18
  NOTE_D5,4, NOTE_E5,8, NOTE_CS5,-8, NOTE_B4,16, NOTE_A4,8, NOTE_A4,8,
NOTE_A4,8,
```

```
  NOTE_E5,4, NOTE_D5,2,};

  int tempo=114;
  int notes = sizeof(melody) / sizeof(melody[0]) / 2;

for (int thisNote = 0; thisNote < notes * 2; thisNote = thisNote + 2)
 {
divider = melody[thisNote + 1];
    if (divider > 0) {
 noteDuration = (wholenote) / divider;
}
  else if (divider < 0)
  {
    noteDuration = (wholenote) / abs(divider);
      noteDuration *= 1.5;
  }
   tone(buzzer, melody[thisNote], noteDuration*0.9);
 delay(noteDuration);
   noTone(buzzer);
}
}
else if(num==6)
{
  int tempo = 200;
  int melody[]={

  NOTE_G4,-8, NOTE_E5,-8, NOTE_G5,-8, NOTE_A5,4, NOTE_F5,8, NOTE_G5,8,
  REST,8, NOTE_E5,4,NOTE_C5,8, NOTE_D5,8, NOTE_B4,-4,
  NOTE_C5,-4, NOTE_G4,8, REST,4, NOTE_E4,-4, // repeats from 3
  NOTE_A4,4, NOTE_B4,4, NOTE_AS4,8, NOTE_A4,4,
  NOTE_G4,-8, NOTE_E5,-8, NOTE_G5,-8, NOTE_A5,4, NOTE_F5,8, NOTE_G5,8,
  REST,8, NOTE_E5,4,NOTE_C5,8, NOTE_D5,8, NOTE_B4,-4, };

  int notes = sizeof(melody) / sizeof(melody[0]) / 2;

for (int thisNote = 0; thisNote < notes * 2; thisNote = thisNote + 2)
 {
divider = melody[thisNote + 1];
    if (divider > 0) {
 noteDuration = (wholenote) / divider;
}
  else if (divider < 0)
  {
    noteDuration = (wholenote) / abs(divider);
```

```
      noteDuration *= 1.5;
  }
   tone(buzzer, melody[thisNote], noteDuration*0.9);
 delay(noteDuration);
   noTone(buzzer);
}
}
else if(num==7)
{
  int tempo = 120;
  int melody[]={
  NOTE_DS4,8,
  NOTE_E4,-4, REST,8, NOTE_FS4,8, NOTE_G4,-4, REST,8, NOTE_DS4,8,
  NOTE_E4,-8, NOTE_FS4,8,  NOTE_G4,-8, NOTE_C5,8, NOTE_B4,-8, NOTE_E4,8,
NOTE_G4,-8, NOTE_B4,8,
  NOTE_AS4,2, NOTE_A4,-16, NOTE_G4,-16, NOTE_E4,-16, NOTE_D4,-16,
  NOTE_E4,2, REST,4, REST,8,
 };

   int notes = sizeof(melody) / sizeof(melody[0]) / 2;

for (int thisNote = 0; thisNote < notes * 2; thisNote = thisNote + 2)
 {
divider = melody[thisNote + 1];
    if (divider > 0) {
 noteDuration = (wholenote) / divider;
}
  else if (divider < 0)
  {
    noteDuration = (wholenote) / abs(divider);
      noteDuration *= 1.5;
  }
   tone(buzzer, melody[thisNote], noteDuration*0.9);
 delay(noteDuration);
   noTone(buzzer);
}
}
else if(num==8)
{
  int melody[]= {
 NOTE_C5,4, //1
  NOTE_F5,4, NOTE_F5,8, NOTE_G5,8, NOTE_F5,8, NOTE_E5,8,
  NOTE_D5,4, NOTE_D5,4, NOTE_D5,4,
  NOTE_G5,4, NOTE_G5,8, NOTE_A5,8, NOTE_G5,8, NOTE_F5,8,
```

```
  NOTE_E5,4, NOTE_C5,4, NOTE_C5,4,
  NOTE_A5,4, NOTE_A5,8, NOTE_AS5,8, NOTE_A5,8, NOTE_G5,8,
  NOTE_F5,4, NOTE_D5,4, NOTE_C5,8, NOTE_C5,8,
  NOTE_D5,4, NOTE_G5,4, NOTE_E5,4,

  };
  int notes = sizeof(melody) / sizeof(melody[0]) / 2;

for (int thisNote = 0; thisNote < notes * 2; thisNote = thisNote + 2)
 {
divider = melody[thisNote + 1];
    if (divider > 0) {
 noteDuration = (wholenote) / divider;
}
  else if (divider < 0)
  {
    noteDuration = (wholenote) / abs(divider);
      noteDuration *= 1.5;
  }
   tone(buzzer, melody[thisNote], noteDuration*0.9);
 delay(noteDuration);
   noTone(buzzer);
}
  }
  else if(num==9)
{
  int melody[]= {
 NOTE_E4,4,  NOTE_E4,4,  NOTE_F4,4,  NOTE_G4,4,//1
  NOTE_G4,4,  NOTE_F4,4,  NOTE_E4,4,  NOTE_D4,4,
  NOTE_C4,4,  NOTE_C4,4,  NOTE_D4,4,  NOTE_E4,4,
  NOTE_E4,-4, NOTE_D4,8,  NOTE_D4,2,
  };
  int tempo=114;
  int notes = sizeof(melody) / sizeof(melody[0]) / 2;

for (int thisNote = 0; thisNote < notes * 2; thisNote = thisNote + 2)
 {
divider = melody[thisNote + 1];
    if (divider > 0) {
 noteDuration = (wholenote) / divider;
}
  else if (divider < 0)
  {
    noteDuration = (wholenote) / abs(divider);
```

```
      noteDuration *= 1.5;
  }
   tone(buzzer, melody[thisNote], noteDuration*0.9);
 delay(noteDuration);
   noTone(buzzer);
}
  }
   else if(num==10)
{
  int melody[]= {

  NOTE_A4,4, NOTE_A4,4, NOTE_A4,4, NOTE_F4,-8, NOTE_C5,16,

  NOTE_A4,4, NOTE_F4,-8, NOTE_C5,16, NOTE_A4,2,//4
  NOTE_E5,4, NOTE_E5,4, NOTE_E5,4, NOTE_F5,-8, NOTE_C5,16,
  NOTE_A4,4, NOTE_F4,-8, NOTE_C5,16, NOTE_A4,2,

  };
  int tempo=120;
  int notes = sizeof(melody) / sizeof(melody[0]) / 2;

for (int thisNote = 0; thisNote < notes * 2; thisNote = thisNote + 2)
 {
divider = melody[thisNote + 1];
    if (divider > 0) {
 noteDuration = (wholenote) / divider;
}
  else if (divider < 0)
  {
    noteDuration = (wholenote) / abs(divider);
      noteDuration *= 1.5;
  }
   tone(buzzer, melody[thisNote], noteDuration*0.9);
 delay(noteDuration);
   noTone(buzzer);
}
  }
     else if(num==11)
{
    int tempo=40;
  int melody[]= {

NOTE_E5, 16, NOTE_DS5, 16, //1
  NOTE_E5, 16, NOTE_DS5, 16, NOTE_E5, 16, NOTE_B4, 16, NOTE_D5, 16, NOTE_C5, 16,
```

```
  NOTE_A4, -8, NOTE_C4, 16, NOTE_E4, 16, NOTE_A4, 16,
  NOTE_B4, -8, NOTE_E4, 16, NOTE_GS4, 16, NOTE_B4, 16,
  NOTE_C5, 8,  REST, 16, NOTE_E4, 16, NOTE_E5, 16,  NOTE_DS5, 16,

  NOTE_E5, 16, NOTE_DS5, 16, NOTE_E5, 16, NOTE_B4, 16, NOTE_D5, 16, NOTE_C5,
16,//6
  NOTE_A4, -8, NOTE_C4, 16, NOTE_E4, 16, NOTE_A4, 16,
  NOTE_B4, -8, NOTE_E4, 16, NOTE_C5, 16, NOTE_B4, 16,
  NOTE_A4 , 4, REST, 8, //9 - 1st ending
  };

  int notes = sizeof(melody) / sizeof(melody[0]) / 2;

for (int thisNote = 0; thisNote < notes * 2; thisNote = thisNote + 2)
 {
divider = melody[thisNote + 1];
    if (divider > 0) {
 noteDuration = (wholenote) / divider;
}
  else if (divider < 0)
  {
    noteDuration = (wholenote) / abs(divider);
      noteDuration *= 1.5;
  }
   tone(buzzer, melody[thisNote], noteDuration*0.9);
 delay(noteDuration);
   noTone(buzzer);
}
  }
   else if(num==12)
{

  int melody[]= {

NOTE_D5,4, NOTE_G4,8, NOTE_A4,8, NOTE_B4,8, NOTE_C5,8, //1
  NOTE_D5,4, NOTE_G4,4, NOTE_G4,4,
  NOTE_E5,4, NOTE_C5,8, NOTE_D5,8, NOTE_E5,8, NOTE_FS5,8,
  NOTE_G5,4, NOTE_G4,4, NOTE_G4,4,
  NOTE_C5,4, NOTE_D5,8, NOTE_C5,8, NOTE_B4,8, NOTE_A4,8,

  NOTE_B4,4, NOTE_C5,8, NOTE_B4,8, NOTE_A4,8, NOTE_G4,8,//6
  NOTE_FS4,4, NOTE_G4,8, NOTE_A4,8, NOTE_B4,8, NOTE_G4,8,
  NOTE_A4,-2,
```

```
  };

  int notes = sizeof(melody) / sizeof(melody[0]) / 2;

for (int thisNote = 0; thisNote < notes * 2; thisNote = thisNote + 2)
 {
divider = melody[thisNote + 1];
    if (divider > 0) {
 noteDuration = (wholenote) / divider;
}
  else if (divider < 0)
  {
    noteDuration = (wholenote) / abs(divider);
      noteDuration *= 1.5;
  }
   tone(buzzer, melody[thisNote], noteDuration*0.9);
 delay(noteDuration);
   noTone(buzzer);
}
  }
     else if(num==13)
{

  int melody[]= {
NOTE_B4, 16, NOTE_B5, 16, NOTE_FS5, 16, NOTE_DS5, 16, //1
  NOTE_B5, 32, NOTE_FS5, -16, NOTE_DS5, 8, NOTE_C5, 16,
  NOTE_C6, 16, NOTE_G6, 16, NOTE_E6, 16, NOTE_C6, 32, NOTE_G6, -16, NOTE_E6, 8,

  NOTE_B4, 16,  NOTE_B5, 16,  NOTE_FS5, 16,   NOTE_DS5, 16,  NOTE_B5, 32,  //2
  NOTE_FS5, -16, NOTE_DS5, 8,  NOTE_DS5, 32, NOTE_E5, 32,  NOTE_F5, 32,
  NOTE_F5, 32,  NOTE_FS5, 32,  NOTE_G5, 32,  NOTE_G5, 32, NOTE_GS5, 32,  NOTE_A5,
16, NOTE_B5, 8
  };
  int tempo=105;

  int notes = sizeof(melody) / sizeof(melody[0]) / 2;

for (int thisNote = 0; thisNote < notes * 2; thisNote = thisNote + 2)
 {
divider = melody[thisNote + 1];
    if (divider > 0) {
 noteDuration = (wholenote) / divider;
}
  else if (divider < 0)
```

```
  {
    noteDuration = (wholenote) / abs(divider);
      noteDuration *= 1.5;
  }
   tone(buzzer, melody[thisNote], noteDuration*0.9);
 delay(noteDuration);
   noTone(buzzer);
}
  }
    else if(num==14)
{

  int melody[]= {

  NOTE_D4,4, NOTE_FS4,8, NOTE_G4,8, NOTE_A4,4, NOTE_FS4,8, NOTE_G4,8,
  NOTE_A4,4, NOTE_B3,8, NOTE_CS4,8, NOTE_D4,8, NOTE_E4,8, NOTE_FS4,8,
NOTE_G4,8,
  NOTE_FS4,4, NOTE_D4,8, NOTE_E4,8, NOTE_FS4,4, NOTE_FS3,8, NOTE_G3,8,
  NOTE_A3,8, NOTE_G3,8, NOTE_FS3,8, NOTE_G3,8, NOTE_A3,2,
  NOTE_G3,4, NOTE_B3,8, NOTE_A3,8, NOTE_G3,4, NOTE_FS3,8, NOTE_E3,8,
  NOTE_FS3,4, NOTE_D3,8, NOTE_E3,8, NOTE_FS3,8, NOTE_G3,8, NOTE_A3,8,
NOTE_B3,8,

  };
  int tempo=100;

  int notes = sizeof(melody) / sizeof(melody[0]) / 2;

for (int thisNote = 0; thisNote < notes * 2; thisNote = thisNote + 2)
 {
divider = melody[thisNote + 1];
    if (divider > 0) {
 noteDuration = (wholenote) / divider;
}
  else if (divider < 0)
  {
    noteDuration = (wholenote) / abs(divider);
      noteDuration *= 1.5;
  }
   tone(buzzer, melody[thisNote], noteDuration*0.9);
 delay(noteDuration);
   noTone(buzzer);
}
  }
```

```
}

 }
```