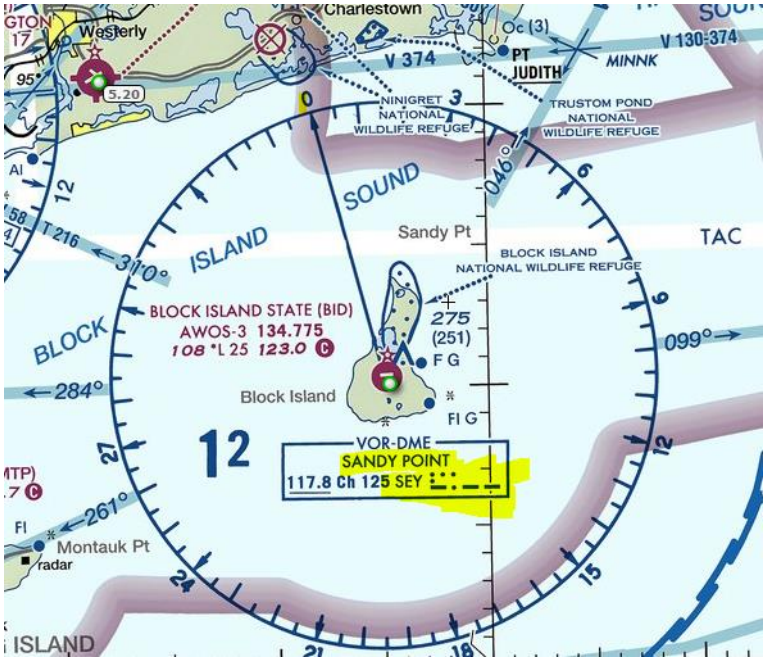
LAB: Morse CODE

Morse code is a signal where combinations of long and short signal (e.g., sounds) are used to encode an alphabet or numbers. It still in use today, for example, aircraft pilots listen to Morse code to identify VOR stations. The aviation map [1] on the right shows a VOR [2] station called Sandy Point and it is located on Block Island. There are other VOR stations nearby; to correctly identify the SANDY POINT station a pilot will tune to the station and then listen for its three letter abbreviation in Morse code, in this case, SEY. The code will sound like:



Beep, beep, beep, beep, long beep, beep, long beep, long beep.

Here is a practice web-site …. I recommend adjusting the speed to 10 wpm

http://morsecode.scphillips.com/translator.html

In this lab you are junior software engineer assigned to fix and complete development of an APP to train pilots in Morse code, the company is depending on this project.
The APP has a problem:

It only converts some typed text into Morse code; for example SOS would be converted to **…** **---** **…** and it will also sounds-out the dots and dashes, but NOT very well. You must add the additional functionality so that the program converts any text (and digitals) into its correct Morse code, complete with sound. While there may be programming syntax that you are not familiar and may not yet understand how the entire program operates, step-by-step, it is still possible for you to recognize a pattern and complete this assignment via research with intelligent trial-and-error.

Ref:
[1] www.skyvector.com
[2] https://en.wikipedia.org/wiki/VHF_omnidirectional_range

LAB: Morse CODE

There are three steps to completing this assignment, you MUST complete all three.

1) Complete the code by:
    a. Enter the program (see next page) and get it working as a baseline;
    b. Fill in the comment heading information with your name, contact, and copyright information;
    c. Use the Internet to find the missing additional Morse Code;
    d. Identify the programming pattern and include the missing Morse code;
    e. Tweak the Beep and Sleep functions to improve the dot-dash sounds (trial and error works).
    f. Add additional comments.
    g. Make certain that your code works using, say, the
        i. Holoalphabetic sentence: *The quick brown fox jumps over the lazy dog*
        ii. Numbers: *0123456789*
    h. Improve the user experience somehow.

2) Research the creation of Morse code and summarize:
    a. Motivation for its development;
    b. What was old or new about this?
    c. Was it easy to find investors? give a reason;
    d. Technical problems that had to be overcome;
    e. Impact on the world.

3) Upload two text files to blackboard before the due date; otherwise, the company will lose a contact and you are out of work.
    a. The complete code: morseAPP.cpp (must be a .cpp asci file)
    b. Your Morse Code research: yourName.pdf (must be a pdf file)

LAB: Morse CODE

```
// © copyright 2010  John Doe.  All Rights Reserved.
// Author: John Doe
// Date:  10-11-2150
// Contact:  john.Doe@somewhere.com
// Description:

#include <iostream>
#include <windows.h>                    // access Sleep(), Beep(),  SetConsoleTitle() … library functions
#include <cstring>                      // string type access.
using namespace std;
//  function prototype
void morseCode(char);

void dot()  {
   Beep(400, 100);                              // sound of a DOT
   cout << '.';                                 // print a DOT
   Sleep(1000);                                 // wait for 1ooo milliseconds
}

void dash()  {
   Beep(500, 200);                      // sound of a DASH
   cout << '-';                         // print a DASH
  Sleep(1000);                          // wait 1ooo milliseconds
 }

int main ()  {
   string in;                 // from  cstring

   cout<< "enter a sentence: ";
   getline(cin, in, '\n');
```

LAB: Morse CODE

```
  int len = in.length();            // get length of string.

  for(int i=0;  i<len;  i++)     {
      char letter = in[i];                        // get ith character from string.
      cout << "letter "<< letter;                 // display character
      morseCode( letter);                         // convert to Morse code sound and dot-dash print
      cout << endl;                               // skip a line on output
      Sleep(1000);                          // wait (about) 1ooo milliseconds
   }
 return 0;  }   // end of main

// INPUT: character
// OUTPUT: Morse Code sounds and dot-dash print
void morseCode(char letter)                       // look for pattern in here …
{
 switch(          toupper(letter)) {         // change character into its Morse code sounds.

  // HERE IS WHERE YOU SHOULD LOOK for a PATTERN
    case 'O':
          dash();  dash();  dash();
          break;
    case 'S':
           dot();  dot();  dot();
           break;
    case ' ':
          // blank space, need a pause without sound.
          break;

    default:
          cout<<"bad input"<< endl;
  }
}    // end of Morse code
```