1. My first draft of the code I had set up a loop that reads through the string, but the parameters where met with using cdstring.at(i) =='R') && (cdstring.at(i-1) != 'O’, this was following the logic that the character at pos 0 was compared to the character at pos 0-1. I realized that debugging this would be painful and tiresome, and so I wrote out a timeline for the string, this allowed me to in co-operate default states which would be updated as the loop reads through the string. All of this made the bool isValidCDPlayerString(string cdstring) to work. I wasn’t able to make sure that the code was in sequential order as I ran out of time.
2. **bool isValidCDPlayerString(string cdstring) works based on steady bool states which update based on the character and the parameter. The other functions call bool isValidCDPlayerString(string cdstring) and work with if loops.**

**c)**

//assert(isValidCDPlayerString("OICP147S") == false );

//assert(isValidCDPlayerString("OICP123SOCP123S") == false );

//assert(isValidCDPlayerString("OICP123P123456SOR") == false );

//assert(isValidCDPlayerString("OICS123OC") == false );

These asserts failed as I wasn’t able to tackle the sequential numbering or the continuation of the tracks if a new cd has not been inserted, or tracks being played after a stop

assert(isValidCDPlayerString("OICPO") == true );

assert(isValidCDPlayerString("OICP123SP456SORICP123") == true );

assert(isValidCDPlayerString("OIRICP123SOR") == true );

assert(isValidCDPlayerString("OICP123ORICP123") == true );

assert(isValidCDPlayerString("OICP123SORICP123456SOR") == true );

assert(isValidCDPlayerString("OICP123OC") == true );

assert(isValidCDPlayerString("OICP123OCP") == true );

assert(isValidCDPlayerString("OICP123OCP456") == true );

these are to test if the string is valid and all of them were true

assert(isOpen("OIRICP123SOR") == true );

assert(isOpen("OICP123ORICP123") == false );

assert(isOpen("OICPO") == true );

assert(isOpen("OICP123O") == true );

assert(isOpen("OICP123OCP456") == false );

assert(isOpen( "OICP123SORICP123456SOR" ) == true );

assert(isOpen( "OICPP123SOR" ) == true );

assert(isOpen( "OICP123OCP" ) == false );

assert(isOpen( "OCOICO" ) == true );

assert(isOpen( "OICP123OS" ) == true );

assert(isOpen( "OICP123ORCS" ) == false );

assert(isOpen( "OICS" ) == false );

to test the isOpen function

assert(hasCD( "OCOICC" ) == false );

assert(hasCD( "OICP123SORICP123456SOR" ) == false );

assert(hasCD( "OICP1P23SOR" ) == false );

assert(hasCD( "OICP123O" ) == true );

assert(hasCD( "OCP123" ) == false );

to test the hasCD function

assert(totalTracksPlayed( "OICP123SP456SORICP123" ) == 9 );

assert(totalTracksPlayed( "OIRICP123SOR" ) == 3 );

assert(totalTracksPlayed( "OICP123ORS" ) == 3 );

assert(totalTracksPlayed( "OCIP123S" ) == -1 );

to test the totalTracksPlayed function

// assert(currentTrack( "OICP123SP456SORICP123" ) == 4);

// assert(currentTrack( "OIRICP123SOR" ) == 4 );

// assert(currentTrack( "OICP123ORS" ) == 4 );

//assert(currentTrack( "OCIP123S" ) == -1 );

I was not able to work out the logic of the currentTrack in time and therefore the asserts failed I was able to set a local variable nextSongTN which kept track of the next song but i think converting the digit into a charcter with the ACSII chart is what messed me up.