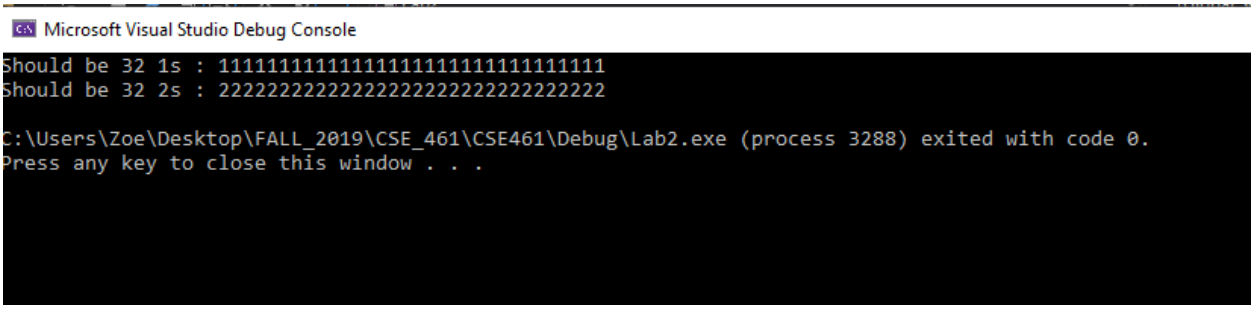
Zoe Veale
CSE 461
Lab 2




```
/*main.cpp*/
#include <iostream>
#include "sDisk.h"

int main(int argc, char* argv) {
  Sdisk disk1("test1", 16, 32);
  std::string block1, block2, block3, block4;
  for (int i = 1; i <= 32; i++) block1 = block1 + "1";
  for (int i = 1; i <= 32; i++) block2 = block2 + "2";
  disk1.PutBlock(4, block1);
  disk1.GetBlock(4, block3);
  std::cout << "Should be 32 1s : ";
  std::cout << block3 << std::endl;
  disk1.PutBlock(8, block2);
  disk1.GetBlock(8, block4);
  std::cout << "Should be 32 2s : ";
  std::cout << block4 << std::endl;;
  return 0;
}

/*sDisk.h*/

#ifndef SDISK_H
#define SDISK_H

#include <string>
#include <fstream>
#include <cstdio>

class Sdisk {
public:
  Sdisk(std::string diskname, int numberofblocks, int blocksize) :
    diskname(diskname), numberofblocks(numberofblocks), blocksize(blocksize) {
    if (!LoadDisk()) {
      CreateDisk();
    }
  }
  int GetBlock(int blocknumber, std::string& buffer) {
    std::ifstream file(diskname.c_str(), std::fstream::binary || std::fstream::app);
    file.seekg((__int64)blocknumber * (__int64)blocksize);

    char* block = new char[blocksize + 1];
    block[blocksize] = '\0';
```

```
    file.read(block, blocksize);
    buffer = std::string(block);
    delete[] block;

    if (!file.good())
      return 0;

    return 1;
  }
  int PutBlock(int blocknumber, std::string buffer) {
    std::ofstream file(diskname.c_str(), std::fstream::binary || std::fstream::out);
    file.seekp((__int64)blocknumber * (__int64)blocksize);
    file.write(buffer.c_str(), blocksize);

    if (!file.good())
      return 0;

    return 1;
  }
  int GetNumberOfBlocks() { return numberofblocks; }
  int GetBlockSize() { return blocksize; }
private:
  std::ifstream inFile;
  std::string diskname;        // file name of software-disk
  int numberofblocks;     // number of blocks on disk
  int blocksize;          // block size in bytes
  bool LoadDisk() {
   return inFile.is_open();
  }
  void CreateDisk() {
    std::ofstream file(diskname.c_str(), std::fstream::binary);
    std::string buffer(numberofblocks * blocksize, '#');
    file.write(buffer.c_str(), (__int64)numberofblocks * (__int64)blocksize);

    if (!file.good())
      printf("File did not write!\n");

  }
};
#endif // !SDISK_H
```