上級プログラミング第三回レポート課題

2019/06/08

us162039　梶田悠

2019

目次

[レポートの課題内容 1](#_Toc9129866)

[補足 2](#_Toc9129867)

[コード 3](#_Toc9129868)

[考察 7](#_Toc9129869)

[入出力演算子 7](#_Toc9129870)

[四則演算子 7](#_Toc9129871)

[const修飾子 9](#_Toc9129872)

# レポートの課題内容

　2 つの入力音声ファイルを結合し、出力音声ファイルに出力するプログラムを作成しなさい。具体的には、-P で指定された音声ファイルの開始サンプル数から終了サンプル数を outputdata の speechdata にコピーし、-F で指定された音声ファイルの開始サンプル数から終了サンプル数を outputdata の speechdata の続きにコピーした音声データを作成し、-O で指定された音声ファイルに出力するようにしなさい。レポート本体、作成したプログラムおよび「report3 -P nitech\_jp\_atr503\_m001\_a01.raw 0 19410 -F nitech\_jp\_atr503\_m001\_a04.raw 52063 70466 -O output.raw」を実行し、プロンプトで「sox -t raw -r 16000 -e signed-integer -b 16 -c 1 output.raw output.wav」を実行した結果得られるoutput.wav を担当教員に提出しなさい。

## 補足

コードは[github](https://github.com/HarukaKajita/jpro_report03)にアップロードしてあるので、レポートのコードが読みづらい場合は適宜参照して下さい。

# コード

ソースコード全文を以下に掲載する。可読性の為にコメントを意図的に削除している部分があります。

コード 1　Data.h

|  |
| --- |
| #include <iostream>  #include <fstream>  #include <cstdlib>  using namespace std;  class Data {  string speechfilename; //データを格納した元の入力ファイル名  short\* speechdata; //データの先頭ポインタ  int samplesize; //データ数  public:  Data() { speechfilename = ""; speechdata = NULL; samplesize = 0; }  void readrawfile(const string&);  void writerawfile(const string&) const;  void memoryalloc(const int&);  void copydata(const Data&, const int&, const int&);  void catdata(const Data&, const int&, const int&);  }; |

コード 2　Data.cpp

|  |
| --- |
| #include "Data.h"  //filename からデータを読み speechdata に格納  void Data::readrawfile(const string& filename) {  speechfilename = filename;  ifstream fin(filename.c\_str(), ios::binary); //バイナリ形式でfilenameを開く  if (!fin) { //エラー処理  cerr << "エラー：ファイルを開けません" << filename << "\n";  exit(EXIT\_FAILURE);  }  fin.seekg(0, ios\_base::end);  int samplesuu = fin.tellg() / sizeof(short);  memoryalloc(samplesuu);  fin.seekg(0, ios\_base::beg);  fin.read((char\*)speechdata, sizeof(short) \* samplesuu)  }  //speechdataの格納データをfilenameにバイナリ形式で書き込む  void Data::writerawfile(const string& filename) const {  ofstream fout(filename.c\_str(), ios::binary); //バイナリ形式でfilenameを開く  if (!fout) { //エラー処理  cerr << "エラー：ファイルを開けません" << filename << "\n";  exit(EXIT\_FAILURE);  }    fout.write((char\*)speechdata, sizeof(short) \* samplesize);  }  //speechdata のメモリを、short で memorysize 分確保する  void Data::memoryalloc(const int& memorysize) {  delete[] this->speechdata;  this->speechdata = new short[memorysize];  }  //第 1 引数で指定された Data クラスのオブジェクト copymoto のデータメンバ speechdata[start]から  //speechdata[end-1]までを、呼び出し元の speechdata[0]から speechdata[end-start-1]へコピーする。  void Data::copydata(const Data& copymoto, const int& start, const int& end) {  this->samplesize = 0;  for (int i = start; i < end; i++) {  this->speechdata[samplesize] = copymoto.speechdata[i];  this->samplesize++;  }  }  //第 1 引数で指定された Data クラスのオブジェクト catmoto のデータメンバspeechdata[start]から  //speechdata[end-1]までを、speechdata[samplesize]以降に追加する。  void Data::catdata(const Data& catmoto, const int& start, const int& end) {  for (int i = start; i < end; i++) {  this->speechdata[samplesize] = catmoto.speechdata[i];  this->samplesize++;  }  } |

コード jpro\_report03.cpp

|  |
| --- |
| #include "Data.h"  #include <vector>  //第二引数の配列から'-'で始まる要素を探し、該当要素のインデックスを第三引数に詰める。該当要素の個数を返す。  void searchOptionArg(const int& size, char\* argv[], vector<int>& indices);  int main(int argc, char\* argv[]) {  char\* inputfilename1 = NULL; //接続前部の音声ファイル名  char\* inputfilename2 = NULL; //接続後部の音声ファイル名  char\* outputfilename = NULL; //出力ファイル名  int startsamplesuu1 = 0; //接続前部を利用する際の開始音声サンプル数  int endsamplesuu1 = 0; //接続前部を利用する際の終了音声サンプル数(この値は含まず)  int startsamplesuu2 = 0; //接続後部を利用する際の開始音声サンプル数  int endsamplesuu2 = 0; //接続後部を利用する際の終了音声サンプル数(この値は含まず)  string filename; //string に変換するための一時ファイル  Data inputdata1; //接続前部の音声ファイルのデータの格納場所  Data inputdata2; //接続後部の音声ファイルのデータの格納場所  //コマンドライン引数の数が不適切な場合は異常終了  if (argc != 11) {  cerr << "Error : Please set args" << endl;  exit(EXIT\_FAILURE);  }  //  const int requiredOptionNum = 3;  vector<int> optionIndices;  searchOptionArg(argc, argv, optionIndices);  #ifdef \_DEBUG  cout << "option Num : " << optionIndices.size() << endl;;  #endif // DEBUG  //オプションが足らない場合は異常終了  if (optionIndices.size() != requiredOptionNum) {  cerr << "Error : Need -P & -F & -O options" << endl;  exit(EXIT\_FAILURE);  }  //コマンドライン引き数の各値を変数に格納  for (int i = 0; i < requiredOptionNum; i++) {  int index = optionIndices.at(i);  switch (argv[index][1])  {  case 'P':  inputfilename1 = argv[index + 1];  startsamplesuu1 = atoi(argv[index + 2]);  endsamplesuu1 = atoi(argv[index + 3]);  break;  case 'F':  inputfilename2 = argv[index + 1];  startsamplesuu2 = atoi(argv[index + 2]);  endsamplesuu2 = atoi(argv[index + 3]);  break;  case 'O':  outputfilename = argv[index + 1];  break;  default:  //不適切なオプションがあれば異常終了  cerr << "Error : Found incorrect option = " << argv[index][1] << endl;  exit(EXIT\_FAILURE);  break;  }  }  #ifdef \_DEBUG  cout << "inputfilename1 : " << inputfilename1 << endl;  cout << "startsamplesuu1 : " << startsamplesuu1 << endl;  cout << "endsamplesuu1 : " << endsamplesuu1 << endl;    cout << "inputfilename2 : " << inputfilename2 << endl;  cout << "startsamplesuu2 : " << startsamplesuu2 << endl;  cout << "endsamplesuu2 : " << endsamplesuu2 << endl;    cout << "outputfilename : " << outputfilename << endl;  #endif // DEBUG  //1 つ目の入力ファイルの読み込み  filename = inputfilename1;  inputdata1.readrawfile(filename);  //2 つ目の入力ファイルの読み込み  filename = inputfilename2;  inputdata2.readrawfile(filename);  //音声データの接続  Data outputdata; //出力ファイルのデータの格納場所  if ((inputfilename1 != NULL) && (startsamplesuu1 != endsamplesuu1) && (inputfilename2 != NULL) &&  (startsamplesuu2 != endsamplesuu2)) {  //出力データを格納するためのメモリを確保する  outputdata.memoryalloc(endsamplesuu1 - startsamplesuu1 + endsamplesuu2 - startsamplesuu2);  //inputdata1 の speechdata の 0 から 19409 までを、  //outputdata の speechdata の 0 から 19409 にコピーする。  outputdata.copydata(inputdata1, startsamplesuu1, endsamplesuu1);  //inputdata2 の speechdata の 52063 から 70466 までを、  //outputdata の speechdata の最後(=この場合 19410)から 18404 個コピーする。  outputdata.catdata(inputdata2, startsamplesuu2, endsamplesuu2);  }  else {  cerr << "音声を接続する前部・後部の音声ファイル名および開始サンプル数・終了サンプル数を指定してください¥n";  exit(EXIT\_FAILURE);  }  //変更したデータをファイルに保存  if (outputfilename != NULL) {  filename = outputfilename;  outputdata.writerawfile(filename);  }  else {  cerr << "outputfilename を指定してください¥n";  exit(EXIT\_FAILURE);  }  return 0;  }  void searchOptionArg(const int& size, char\* argv[], vector<int>& indices)  {  for (int i = 0; i < size; i++) {  if (argv[i][0] == '-') {  indices.push\_back(i);  };  }  } |

# 実行結果

　上記のプログラムを実行して得られたoutput.rawに対して、指定されたsoxコマンドを実行して生成されたoutput.wavファイルを添付しているのでそちらを適宜参照して下さい。

# 考察

## 入出力演算子