計算機程式設計 HW9report

103061223 李俊穎

1. Introduction of the problem and your solution

The homework 9 is a “Music Play List” that we have to read the music file name and sort them by using track ID, performer, song title and duration, and if we have others types of files in the directory, we need to exclude them from sorting and output. Because this time most of the codes have already been given, I just need to separate the filename read from directory and store them into a class, and push_back to the vector. I reference the method of separate strings codes in lab8. After that, I write four functions according to four keys to sort the songs. Because strings can be compared with each other directly (by their ASCII code), I don’t need to write another function to compare the magnitude with two strings, and it makes me easier to complete this homework.

2. Implementation details ,Additional features of your program (data structure, flows and algorithms)

In the first step I use the method in lab8 to separate the data and store them into a class:

```
for(int i=0;i<FileNamelist.size();i++)
{
    lastpos=FileNamelist[i].find_first_not_of("-",0);    //
    pos=FileNamelist[i].find_first_of("-",lastpos);
    ID=FileNamelist[i].substr(lastpos,pos-lastpos);
    buffer.setID(ID);
    lastpos=FileNamelist[i].find_first_not_of("-",pos);  /
    pos=FileNamelist[i].find_first_of("-",lastpos);
    performer=FileNamelist[i].substr(lastpos,pos-lastpos);
    buffer.setperformer(performer);
    lastpos=FileNamelist[i].find_first_not_of("-",pos);  /
    pos=FileNamelist[i].find_first_of("-",lastpos);
    songtitle=FileNamelist[i].substr(lastpos,pos-lastpos);
    buffer.setsongtitle(songtitle);
    lastpos=FileNamelist[i].find_first_not_of("-",pos);  /
    pos=FileNamelist[i].find_first_of(".",lastpos);
    duration=FileNamelist[i].substr(lastpos,pos-lastpos);
    buffer.setduration(duration);
    lastpos=FileNamelist[i].find_first_not_of(".",pos);  /
    pos=FileNamelist[i].find_first_of("-",lastpos);
    file=FileNamelist[i].substr(lastpos,pos-lastpos);
    if(file!="mp3") //if the file is not mp3,then execute
    {
        continue;
    }
    buffer.setpath(FullPathlist[i]);
    seperate.push_back(buffer);
}
```

The last paragraph is to check the types of the files. If it is string variable is not

“mp3”, the continue will force it to execute the next loop, and this file won’t be sorted and output.

I choose selection sort to do this homework. Four of the functions are similar like the following codes:

```
void sort_id(vector<newfilename> &seperate)      //use selection sort to sort the II
{
    unsigned int startScan, minIndex;
    newfilename minValue;
    for (startScan=0; startScan<seperate.size()-1; startScan++) {
        minIndex = startScan;
        minValue = seperate[startScan];
        for (unsigned int index=startScan+1; index<seperate.size(); index++) {
            if (seperate[index].getID() < minValue.getID()) {
                minValue = seperate[index];
                minIndex = index;
            }
        }
        seperate[minIndex]  = seperate[startScan];
        seperate[startScan] = minValue;
    }
}
```

By the way, the class I write to store the data is the following:

```
class newfilename   //to store the seperated filename and path that read from file
{
private:
    string ID;
    string performer;
    string songtitle;
    string duration;
    string path;
public:
    void setID(string x){ID=x;}
    void setperformer(string y){performer=y;}
    void setsongtitle(string z){songtitle=z;}
    void setduration(string w){duration=w;}
    void setpath(string s){path=s;}
    string getID(){return ID;}
    string getperformer(){return performer;}
    string getsongtitle(){return songtitle;}
    string getduration(){return duration;}
    string getpath(){return path;}
};
```

Then this homework is finished!!