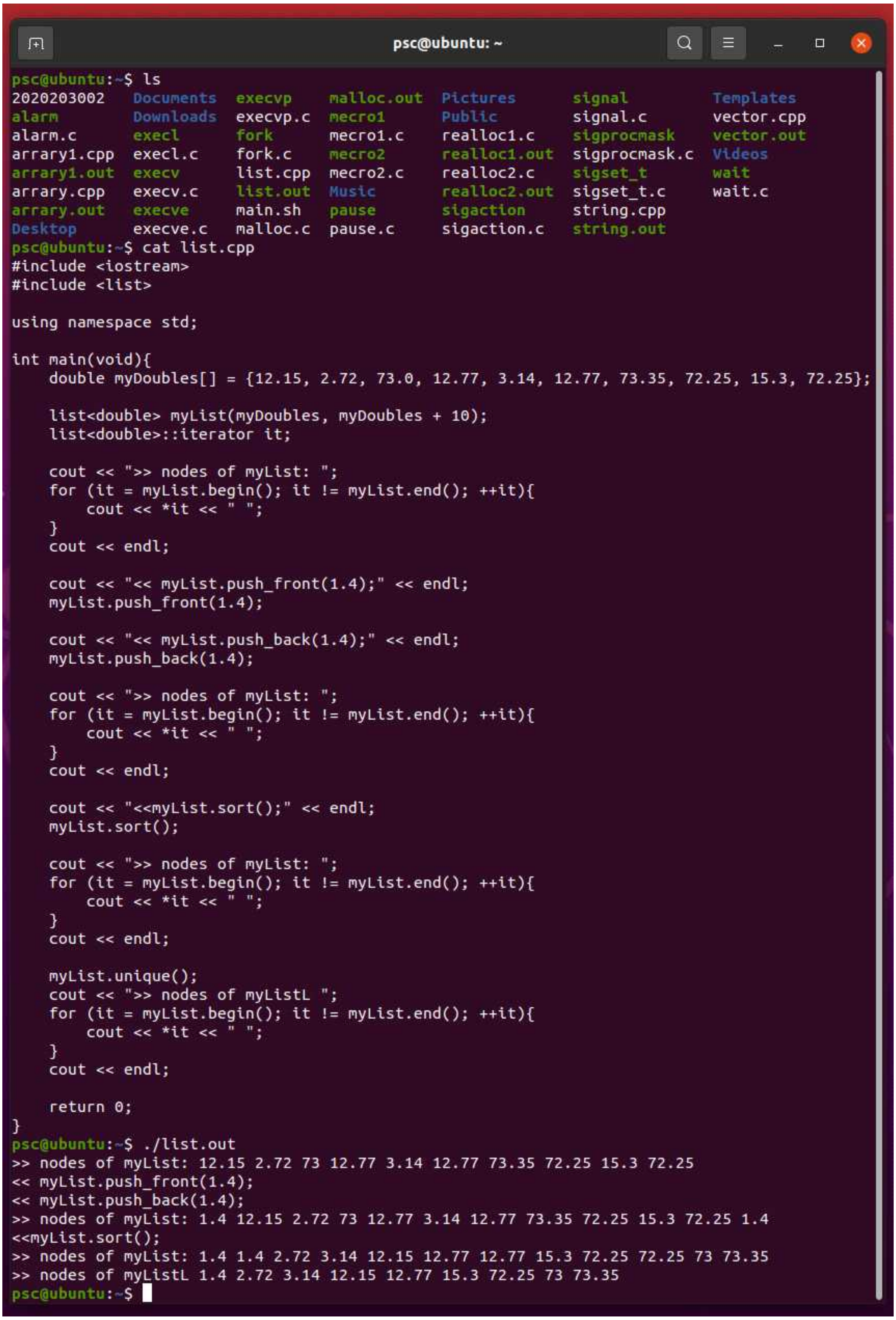
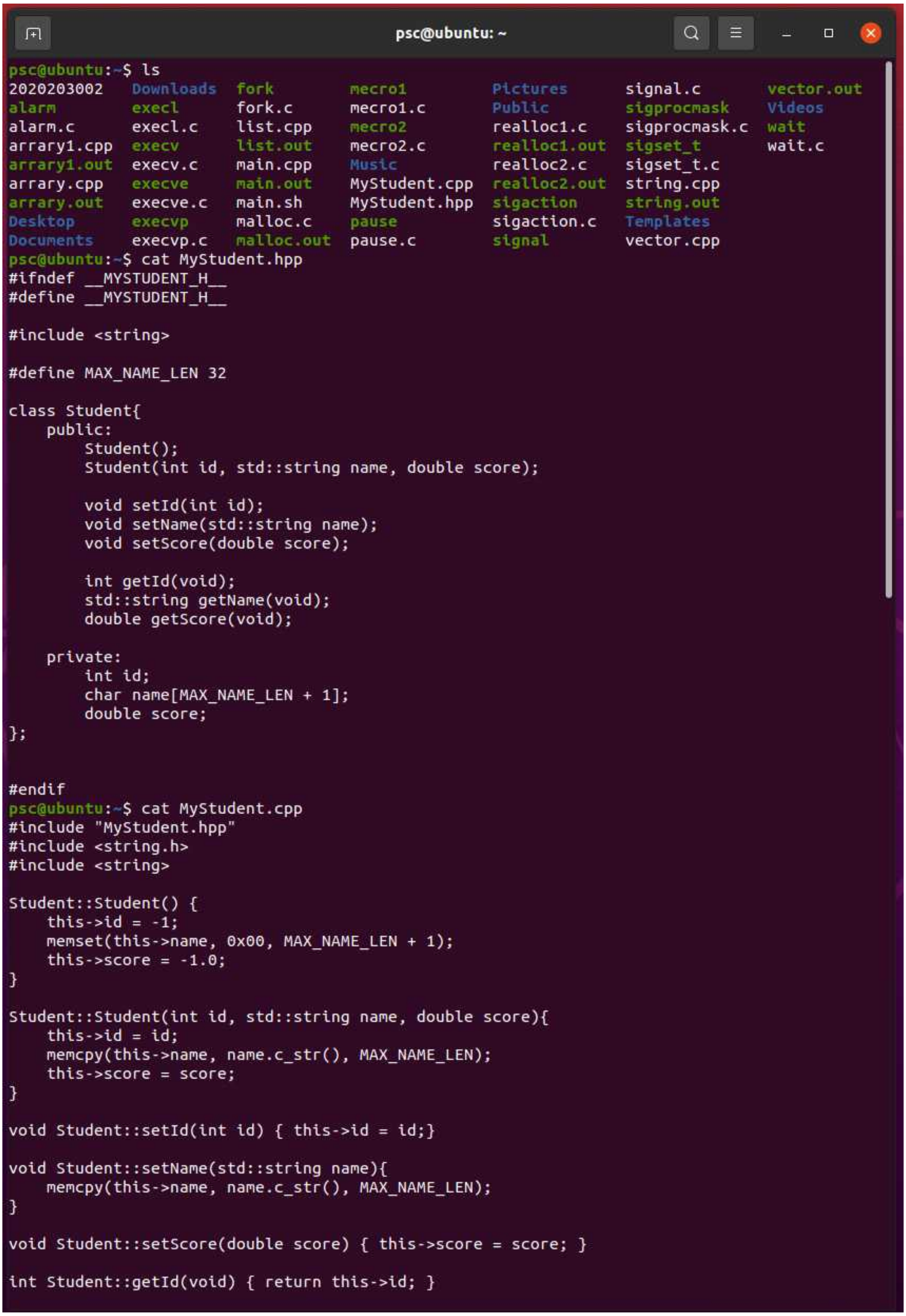
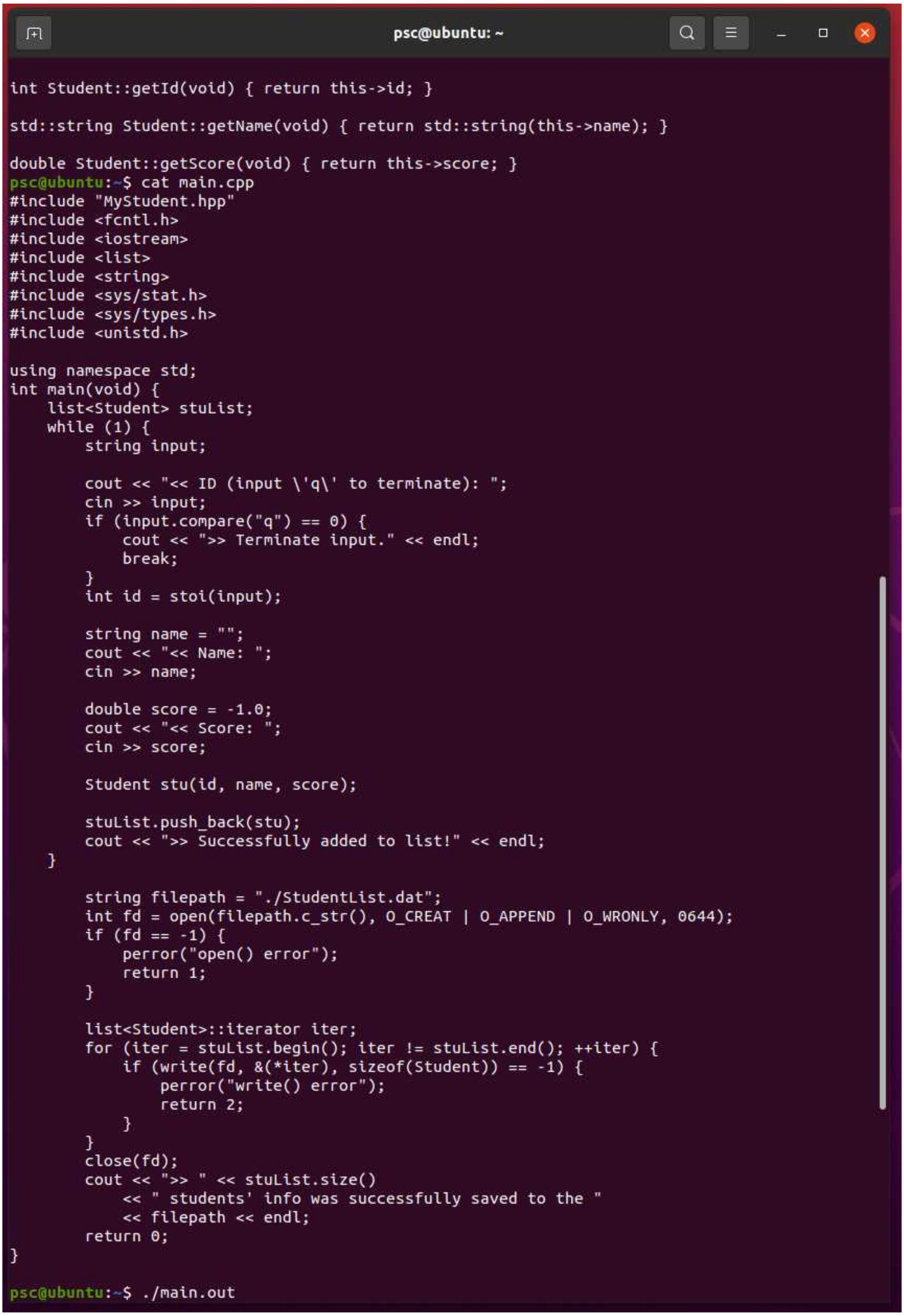
2020203002 박상천



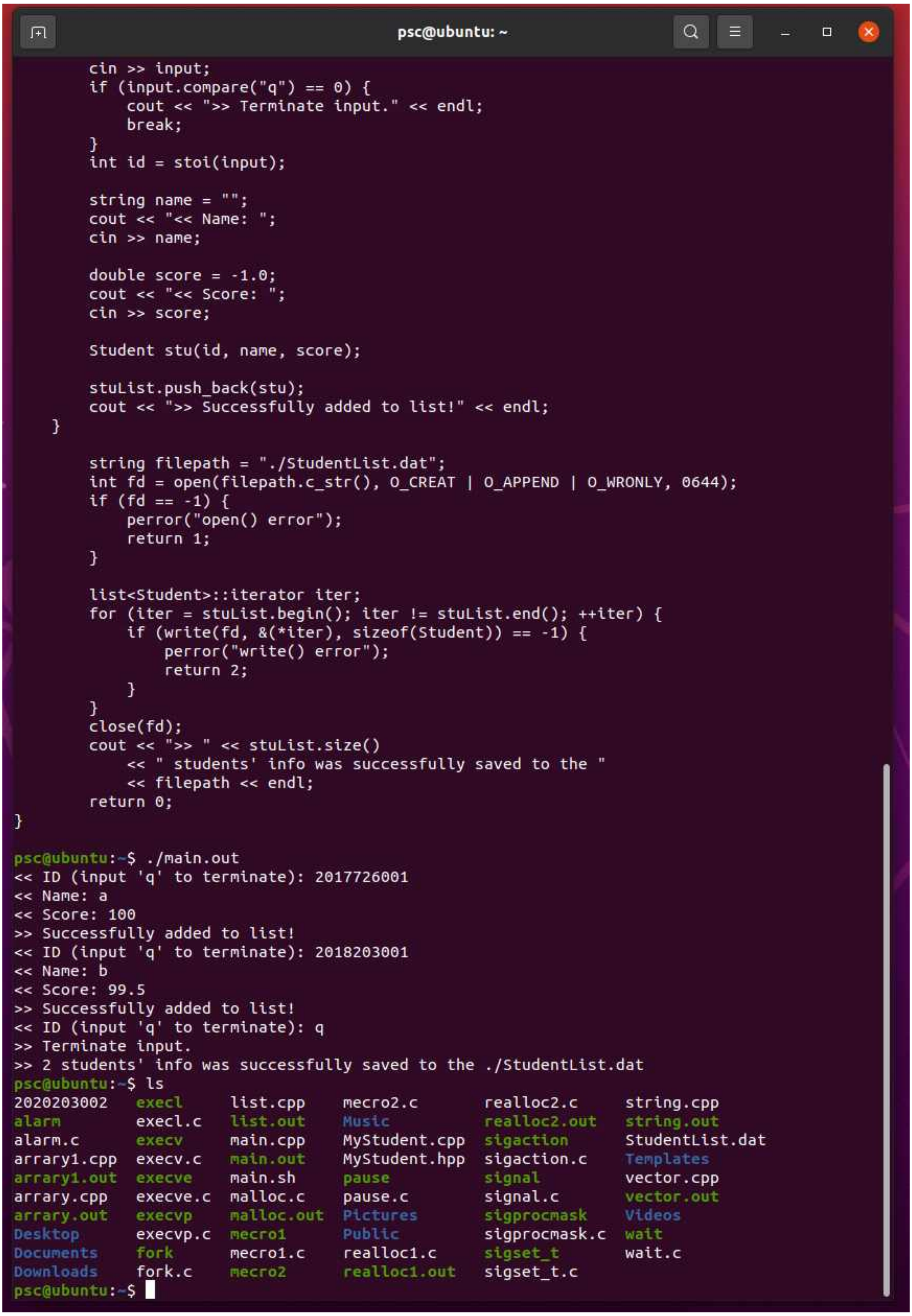
```
psc@ubuntu:~$ ls
2020203002   Documents  execvp    malloc.out  Pictures      signal         Templates
alarm        Downloads  execvp.c  mecro1      Public        signal.c       vector.cpp
alarm.c      execl      fork      mecro1.c    realloc1.c    sigprocmask    vector.out
arrary1.cpp  execl.c    fork.c    mecro2      realloc1.out  sigprocmask.c  Videos
arrary1.out  execv      list.cpp  mecro2.c    realloc2.c    sigset_t       wait
arrary.cpp   execv.c    list.out  Music       realloc2.out  sigset_t.c     wait.c
arrary.out   execve     main.sh   pause       sigaction     string.cpp
Desktop      execve.c   malloc.c  pause.c     sigaction.c   string.out
psc@ubuntu:~$ cat list.cpp
#include <iostream>
#include <list>

using namespace std;

int main(void){
    double myDoubles[] = {12.15, 2.72, 73.0, 12.77, 3.14, 12.77, 73.35, 72.25, 15.3, 72.25};

    list<double> myList(myDoubles, myDoubles + 10);
    list<double>::iterator it;

    cout << ">> nodes of myList: ";
    for (it = myList.begin(); it != myList.end(); ++it){
        cout << *it << " ";
    }
    cout << endl;

    cout << "<< myList.push_front(1.4);" << endl;
    myList.push_front(1.4);

    cout << "<< myList.push_back(1.4);" << endl;
    myList.push_back(1.4);

    cout << ">> nodes of myList: ";
    for (it = myList.begin(); it != myList.end(); ++it){
        cout << *it << " ";
    }
    cout << endl;

    cout << "<<myList.sort();" << endl;
    myList.sort();

    cout << ">> nodes of myList: ";
    for (it = myList.begin(); it != myList.end(); ++it){
        cout << *it << " ";
    }
    cout << endl;

    myList.unique();
    cout << ">> nodes of myListL ";
    for (it = myList.begin(); it != myList.end(); ++it){
        cout << *it << " ";
    }
    cout << endl;

    return 0;
}
psc@ubuntu:~$ ./list.out
>> nodes of myList: 12.15 2.72 73 12.77 3.14 12.77 73.35 72.25 15.3 72.25
<< myList.push_front(1.4);
<< myList.push_back(1.4);
>> nodes of myList: 1.4 12.15 2.72 73 12.77 3.14 12.77 73.35 72.25 15.3 72.25 1.4
<<myList.sort();
>> nodes of myList: 1.4 1.4 2.72 3.14 12.15 12.77 12.77 15.3 72.25 72.25 73 73.35
>> nodes of myListL 1.4 2.72 3.14 12.15 12.77 15.3 72.25 73 73.35
psc@ubuntu:~$ 
```



```
psc@ubuntu:~$ ls
2020203002   Downloads  fork        mecro1         Pictures      signal.c       vector.out
alarm        execl      fork.c      mecro1.c       Public        sigprocmask    Videos
alarm.c      execl.c    list.cpp    mecro2         realloc1.c    sigprocmask.c  wait
arrary1.cpp  execv      list.out    mecro2.c       realloc1.out  sigset_t       wait.c
arrary1.out  execv.c    main.cpp    Music          realloc2.c    sigset_t.c
arrary.cpp   execve     main.out    MyStudent.cpp  realloc2.out  string.cpp
arrary.out   execve.c   main.sh     MyStudent.hpp  sigaction     string.out
Desktop      execvp     malloc.c    pause          sigaction.c   Templates
Documents    execvp.c   malloc.out  pause.c        signal        vector.cpp
psc@ubuntu:~$ cat MyStudent.hpp
#ifndef __MYSTUDENT_H__
#define __MYSTUDENT_H__

#include <string>

#define MAX_NAME_LEN 32

class Student{
    public:
        Student();
        Student(int id, std::string name, double score);

        void setId(int id);
        void setName(std::string name);
        void setScore(double score);

        int getId(void);
        std::string getName(void);
        double getScore(void);

    private:
        int id;
        char name[MAX_NAME_LEN + 1];
        double score;
};


#endif
psc@ubuntu:~$ cat MyStudent.cpp
#include "MyStudent.hpp"
#include <string.h>
#include <string>

Student::Student() {
    this->id = -1;
    memset(this->name, 0x00, MAX_NAME_LEN + 1);
    this->score = -1.0;
}

Student::Student(int id, std::string name, double score){
    this->id = id;
    memcpy(this->name, name.c_str(), MAX_NAME_LEN);
    this->score = score;
}

void Student::setId(int id) { this->id = id;}

void Student::setName(std::string name){
    memcpy(this->name, name.c_str(), MAX_NAME_LEN);
}

void Student::setScore(double score) { this->score = score; }

int Student::getId(void) { return this->id; }
```



```
int Student::getId(void) { return this->id; }

std::string Student::getName(void) { return std::string(this->name); }

double Student::getScore(void) { return this->score; }
psc@ubuntu:~$ cat main.cpp
#include "MyStudent.hpp"
#include <fcntl.h>
#include <iostream>
#include <list>
#include <string>
#include <sys/stat.h>
#include <sys/types.h>
#include <unistd.h>

using namespace std;
int main(void) {
    list<Student> stuList;
    while (1) {
        string input;

        cout << "<< ID (input \'q\' to terminate): ";
        cin >> input;
        if (input.compare("q") == 0) {
            cout << ">> Terminate input." << endl;
            break;
        }
        int id = stoi(input);

        string name = "";
        cout << "<< Name: ";
        cin >> name;

        double score = -1.0;
        cout << "<< Score: ";
        cin >> score;

        Student stu(id, name, score);

        stuList.push_back(stu);
        cout << ">> Successfully added to list!" << endl;
    }

        string filepath = "./StudentList.dat";
        int fd = open(filepath.c_str(), O_CREAT | O_APPEND | O_WRONLY, 0644);
        if (fd == -1) {
            perror("open() error");
            return 1;
        }

        list<Student>::iterator iter;
        for (iter = stuList.begin(); iter != stuList.end(); ++iter) {
            if (write(fd, &(*iter), sizeof(Student)) == -1) {
                perror("write() error");
                return 2;
            }
        }
        close(fd);
        cout << ">> " << stuList.size()
            << " students' info was successfully saved to the "
            << filepath << endl;
        return 0;
}

psc@ubuntu:~$ ./main.out
```



```
        cin >> input;
        if (input.compare("q") == 0) {
            cout << ">> Terminate input." << endl;
            break;
        }
        int id = stoi(input);

        string name = "";
        cout << "<< Name: ";
        cin >> name;

        double score = -1.0;
        cout << "<< Score: ";
        cin >> score;

        Student stu(id, name, score);

        stuList.push_back(stu);
        cout << ">> Successfully added to list!" << endl;
    }

        string filepath = "./StudentList.dat";
        int fd = open(filepath.c_str(), O_CREAT | O_APPEND | O_WRONLY, 0644);
        if (fd == -1) {
            perror("open() error");
            return 1;
        }

        list<Student>::iterator iter;
        for (iter = stuList.begin(); iter != stuList.end(); ++iter) {
            if (write(fd, &(*iter), sizeof(Student)) == -1) {
                perror("write() error");
                return 2;
            }
        }
        close(fd);
        cout << ">> " << stuList.size()
            << " students' info was successfully saved to the "
            << filepath << endl;
        return 0;
}

psc@ubuntu:~$ ./main.out
<< ID (input 'q' to terminate): 2017726001
<< Name: a
<< Score: 100
>> Successfully added to list!
<< ID (input 'q' to terminate): 2018203001
<< Name: b
<< Score: 99.5
>> Successfully added to list!
<< ID (input 'q' to terminate): q
>> Terminate input.
>> 2 students' info was successfully saved to the ./StudentList.dat
psc@ubuntu:~$ ls
2020203002   execl     list.cpp    mecro2.c       realloc2.c     string.cpp
alarm        execl.c   list.out    Music          realloc2.out   string.out
alarm.c      execv     main.cpp    MyStudent.cpp  sigaction      StudentList.dat
arrary1.cpp  execv.c   main.out    MyStudent.hpp  sigaction.c    Templates
arrary1.out  execve    main.sh     pause          signal         vector.cpp
arrary.cpp   execve.c  malloc.c    pause.c        signal.c       vector.out
arrary.out   execvp    malloc.out  Pictures       sigprocmask    Videos
Desktop      execvp.c  mecro1      Public         sigprocmask.c  wait
Documents    fork      mecro1.c    realloc1.c     sigset_t       wait.c
Downloads    fork.c    mecro2      realloc1.out   sigset_t.c
psc@ubuntu:~$
```