HW2: Programming Project #2 (Report Start From Page 2)

Due: May 1st, 23:59

This is the second part of a programming project begun in HW1. We add methods to store object as records in files and load objects from files, using the IOBUffer classes of this chapter.

1. Add Pack and Unpack methods to class **Student**. Use class BufferFile to create a file of students records. Test these methods using the types of buffers supported by the IOBuffer classes.
2. Add Pack and UnPack methods to class **CourseRegistration**. Use class BufferFile to create a file of course registration. Test these methods using the types of buffers supported by the IOBuffer classes.

The next part of the programming projects is in HW3.

Requirements:

- The program should be compiled and executed in the Linux environments.
- Students compile their programs using g++ and make utility
- Please use separate compile (multiple compile)
- *.cpp files for IOBuffer and BufferFile class **should be complied as a shared library**. Study the makefile of the source code repository.
- Test your program by using more than 10 Student objects and 10 CourseRegistration objects, respectively.

What to submit

- Please upload your source codes, Makefile, the report file (soft copy) **to your github repository** with a push command.
- Use an attached file as a report template.

Report Of HW2 File_Structure

학부 : 정보컴퓨터공학부

학번 : 201624548

이름 : 이재윤

■ 목표

Linux (Ubuntu) 시스템에서 C++ 프로그래밍을 할 수 있다.

C++의 클래스 설계와 캡슐화에 익숙하다.

C++의 클래스 상속에 대해 익숙하다.

필드와 레코드에 대해 익숙하다.

■ 내용

Student.h - HW1 의 내용에 Pack, Unpack 함수 추가

Student.cpp - HW1 의 내용에 Pack, Unpack 함수 추가

CourseRegistratio.h - HW1 의 내용에 Pack, Unpack, delStudent, delPointer 함수 추가

CourseRegistratio.cpp - HW1 의 내용에 Pack, Unpack, delStudent, delPointer 함수 추가

iobuffer.h - iobuffer class 선언

iobuffer.cpp - buffile class 함수 바디 구현

buffile.h - buffile class 선언

buffile.cpp - buffile class 함수 바디 구현

■ 내용에 대한 분석

Student.h , Student.cpp

Student.h 파일에는 Pack, Unpack 함수를 선언만 해 주었다.

Student.cpp 파일에는 함수 바디를 구현하였는데 내용은 다음과 같다.

```
int Student::Pack (IOBuffer & Buffer) const
{// return TRUE if all succeed, FALSE o/w
        int numBytes;
        char buff[30];
        Buffer . Clear ();
        // Int 자료형을 저장하기 위하여  sprint 사용
        sprintf(buff,"%d",identifier);
        numBytes = Buffer . Pack (buff);
        if (numBytes == -1) return FALSE;
        numBytes = Buffer . Pack (name);
        if (numBytes == -1) return FALSE;
        numBytes = Buffer . Pack (address);
        if (numBytes == -1) return FALSE;
        sprintf(buff,"%d",(int)first_enrollment);
        numBytes = Buffer . Pack (buff);
        if (numBytes == -1) return FALSE;
        sprintf(buff,"%d",(int)credit_hours);
        numBytes = Buffer . Pack (buff);
        if (numBytes == -1) return FALSE;
        return TRUE;
}
```

Student 클래스에 int형 자료가 존재하기 때문에, 배열로 바꾸어 iobuffer 클래스에 Pack 해 주었다.

Unpack 함수 또한 다음과 같다.

```
int Student::Unpack (IOBuffer & Buffer)
{// unpack with maximum size, and add null termination to strings

        int numBytes;
        char buff[max_addresslength];
        // 배열 초기화
        memset(buff, NULL, max_addresslength);
        numBytes = Buffer . Unpack (buff);
        if (numBytes == -1) return FALSE;
        // Int 자료형으로 저장하기 위해 atoi 사용
        setIdentifier(atoi(buff));

        memset(buff, NULL, max_addresslength);
        numBytes = Buffer . Unpack (buff);
        if (numBytes == -1) return FALSE;
        setName(buff);

        memset(buff, NULL, max_addresslength);
        numBytes = Buffer . Unpack (buff);
        if (numBytes == -1) return FALSE;
        setAddress(buff);

        memset(buff, NULL, max_addresslength);
        numBytes = Buffer . Unpack (buff);
        if (numBytes == -1) return FALSE;
        setFirstEnrollment(atoi(buff));

        memset(buff, NULL, max_addresslength);
        numBytes = Buffer . Unpack (buff);
        if (numBytes == -1) return FALSE;
        setCreditHours(atoi(buff));

        return TRUE;
}
```

버퍼를 초기화 해 주고, atoi 함수와 지난번 선언해 주었던 set 함수들을 이용하여 값을 넣어줬다.

CourseRegistration.h , CourseRegistration.cpp

CourseRegistration.h 파일에는 Pack, Unpack 함수를 선언만 해 주었다.

CourseRegistration.cpp 파일에는 함수 바디를 구현하였는데 학생을 링크드 리스트로 구현하여 코드가 조금 길어졌다.

또한 학생의 제거와 메모리 누수를 방지하기위한 메모리 해제 작업 또한 추가되었다.

추가된 내용을 살펴보자

모든 포인터 해제 작업이다

```
void CourseRegistration::delPointer(void) {
	STD * si = course_grades;
	STD * pre;

	while (si != NULL) {
		pre = si;
		si = si->nxt_std;
		free(pre);
	}
}
```

학생 아이디를 파라미터로 학생을 제거하는 작업이다

```
void CourseRegistration::delStudent(int tmpidentifier) {
	STD * si = course_grades;
	STD * pre;

	int isvalid = 0;
	while (si != NULL) {
		if (si->std_identifier == tmpidentifier) {
			isvalid = 1;
			break;
		}
		pre = si;
		si = si->nxt_std;
	}
	if (isvalid == 1) {
		pre->nxt_std = si->nxt_std;
		free(si);
	}
}
```

또한 아래는 Pack Unpack 함수인데, Student.cpp에서 구현한 함수를 Linked-List에 걸맞게 수정만 조금 하였다.

```
int CourseRegistration::Pack (IOBuffer & Buffer) const
{// return TRUE if all succeed, FALSE o/w
        int numBytes;
        char buff[30];
        Buffer . Clear ();
        sprintf(buff,"%d",course_identifier);
        numBytes = Buffer . Pack (buff);
        if (numBytes == -1) return FALSE;

        sprintf(buff,"%d",class_credit_hours);
        numBytes = Buffer . Pack (buff);
        if (numBytes == -1) return FALSE;

        STD * si = course_grades->nxt_std;
        int index = 1;
        while (si != NULL) {
                sprintf(buff,"%d",si->std_identifier);
                numBytes = Buffer . Pack (buff);
                if (numBytes == -1) return FALSE;

                sprintf(buff,"%d",si->grade_sum);
                numBytes = Buffer . Pack (buff);
                if (numBytes == -1) return FALSE;

                si = si->nxt_std;
        }
        return TRUE;
}
```

```
int CourseRegistration::Unpack (IOBuffer & Buffer)
{// unpack with maximum size, and add null termination to strings

        int numBytes;
        char buff[max_addresslength];
        // 배열 초기화
        memset(buff, NULL, max_addresslength);
        numBytes = Buffer . Unpack (buff);
        if (numBytes == -1) return FALSE;
        setcourse_identifier(atoi(buff));

        memset(buff, NULL, max_addresslength);
        numBytes = Buffer . Unpack (buff);
        if (numBytes == -1) return FALSE;
        setclass_credit_hours(atoi(buff));

        while ( numBytes != -1) {
                int tmpid, tmpgrade;
                memset(buff, NULL, max_addresslength);
                numBytes = Buffer.Unpack(buff);
                if (numBytes == -1)   return TRUE;
                tmpid = atoi(buff);

                memset(buff, NULL, max_addresslength);
                numBytes = Buffer.Unpack(buff);
                if (numBytes == -1) return FALSE;
                tmpgrade = atoi(buff);

                addGrade(tmpid, tmpgrade);
        }

        return TRUE;
}
```

위에 파랗게 if문에 True를 반환하는 이유는, 학생이 0명이던, 여러명이던 학번이 없는 경우라면 그 수업에 학생을 추가하기 위한 필요한 작업은 끝이 났다고 볼 수 있고

그러므로 TRUE를 반환하게 작업하였다.

iobuffer.h , iobuffer.cpp

iobuffer 파일은 부모 클래스이고, 가상함수가 존재하였으나, 파일이 많아지면 불편해지기 때문에 필요한 내용만 추려 가변길이 레코드로 바꾸었다. (Delim = '|' 이용)

중요 함수에 대한 분석을 해 보자 (내용을 복사하기엔 양이 많아 함수의 동작 방식만 간단히 설명하겠다.)

Read()

값을 읽어오는 함수이다. 2바이트의 BufferSize를 읽어오고, BufferSize만큼 그 후의 값을 읽어온다. 읽어온 값은 Buffer에 저장하는데. 이는 나중에 Unpack할 때 사용된다.

Write()

값을 쓰는 함수이다. 버퍼사이즈를 맨 앞에 넣어주고 (Read 할 때 필요한 만큼만 읽기 위함) delimiter를 마지막에 추가해준다.

Pack()

Buffer에 받아온 field의 값들을 저장한다. 마지막엔 delimiter을 추가한다.

에러가 발생하면 -1을 리턴한다.

Unpack()

Buffer 배열의 NextByte 부터 delimiter가 나타나는 장소까지 읽어내어 받아온 포인터에 저장한다. NextByte엔 길이와 1 (delimiter 1개)을 더한다. 에러가 발생하면 -1을 리턴한다.

ReadHeader()

헤더를 읽고, 읽어온 내용을 headerStr과 비교하여 다르다면 -1을 리턴하는 단순한 함수이다.

내가 읽고싶은 파일이 목적 파일이 맞는지 확인하기 위해 사용한다.

WriteHeader()

headerStr을 파일 맨 앞에 쓴다. 이 파일이 내 파일임을 알리기 위한 함수이다.

Clear()

NextByte (Unpack 함수에서 사용)을 초기화한다.

BufferFile.h , BufferFile.cpp

iobuffer 클래스를 중요 함수로 사용하며, 파일 생성, 읽기, 쓰기, 닫기 등등을 하는 클래스이다.

중요 함수에 대한 분석을 해 보자 (내용을 복사하기엔 양이 많아 함수의 동작 방식만 간단히 설명하겠다.)

Create()

파일이름을 받아와 파일을 새로 만든다. 파일을 만들 때 iobuffer에 선언되었던 WriteHeader 함수를 File을 파라미터로 호출하여 파일에 헤더파일을 쓴다.

Open()

파일이름을 받아와 파일을 여는 함수이다. 파일을 읽을 때 iobuffer에 선언되었던 ReadHeader 함수를 File을 파라미터로 호출하여 내가 읽고싶은 파일이 맞는지 확인한다.

Close()

열었던 파일을 닫는다

Rewind()

파일 위치 초기화 함수이다. 헤더 이후로 이동시켜준다.

Read()

내부 변수 iobuffer의 Read함수를 호출하여 버퍼를 읽어온다.

WriteHeader()

내부 변수 iobuffer의 Write함수를 호출하여 버퍼를 쓴다.

Append()

파일의 끝에 내용을 추가해준다.

Driver.cpp

데이터를 생성 및 저장, 불러오기를 하는 드라이버 함수이다.

학생 이름, 사는 지역만 작업하였고 나머지는 모두 랜덤하게 작용한다.

또한 iobuffer의 maxbyte를 넘기지 않기 위해 학생은 5명씩 (중복은 배제된다.) 랜덤으로 삽입했다.

```
// 저장된 데이터 불러오기
Student St[10];
RecordingFile.Open((char*)"Students.dat", ios::in);

i = 0;
while (RecordingFile.Read() != -1) {
		St[i++].Unpack(Buffer);
		cout << St[i-1] << endl;
}

RecordingFile.Close();

CourseRegistration cr[10];
// 수업을 랜덤으로 생성
for (i = 0; i < 10; i++) {
		cr[i].setCourseRegistration(rand(), rand() % 3 + 1);
}
// 학생을 랜덤으로 삽입
for (i = 0; i < 10; i++) {
		for (int j = 0; j < 5; j++){
			cr[i].addstudent(St[rand()%10]);
		}
		cout << cr[i] << endl;
}
// 데이터 파일 저장
remove("CourseRegistration.dat");
RecordingFile.Create ((char*)"CourseRegistration.dat", ios::out);
for (int i = 0; i < 10 ; i++){
		cr[i].Pack(Buffer);
		RecordingFile.Write();
}

RecordingFile.Close();

// 저장된 데이터 불러오 기
```

```
RecordingFile.Open((char*)"CourseRegistration.dat", ios::in);

i = 0;
while (RecordingFile.Read() != -1) {
        cr[i++].Unpack(Buffer);
        cout << cr[i-1] << endl;
}

// 데이터 누수 방지를 위해 포인터 해제
for (int i = 0; i < 10 ; i++){
        cr[i].delPointer();
}


RecordingFile.Close();
```

이상 파일은 모두 살펴보았고, 결과창을 확인하자.

■ 결과

```
lep@lep-Virtual-Machine:~/HW/FIle_Structure/HW2$ ls
CourseRegistration.cpp  Driver.cpp  Student.cpp  buffile.cpp  iobuffer.cpp
CourseRegistration.h    Makefile    Student.h    buffile.h    iobuffer.h
lep@lep-Virtual-Machine:~/HW/FIle_Structure/HW2$ make
g++     -c -o Driver.o Driver.cpp
g++     -c -o Student.o Student.cpp
g++     -c -o CourseRegistration.o CourseRegistration.cpp
g++     -c -o iobuffer.o iobuffer.cpp
g++     -c -o buffile.o buffile.cpp
gcc -o a.out Driver.o Student.o CourseRegistration.o iobuffer.o buffile.o -lstdc
++
lep@lep-Virtual-Machine:~/HW/FIle_Structure/HW2$ ls
CourseRegistration.cpp  Driver.o     Student.o    buffile.o
CourseRegistration.h    Makefile     a.out        iobuffer.cpp
CourseRegistration.o    Student.cpp  buffile.cpp  iobuffer.h
Driver.cpp              Student.h    buffile.h    iobuffer.o
```

```
lep@lep-Virtual-Machine:~/HW/FIle_Structure/HW2$ ./a.out
id : 201624548
name : lee jae yoon
address : Pusan
first_enrollment : 2009-09-14-Mon-01-24-16
credit_hours : 62

id : 62925608
name : KIM
address : Seoul
first_enrollment : 1992-01-01-Wed-16-33-41
credit_hours : 79

id : 1326149140
name : BONG
address : Daejeon
first_enrollment : 2003-04-19-Sat-01-23-14
credit_hours : 72
```

```
Class_Identifier : 367666014
Class_Credit_Hours : 3
 ##### Blow the Students #####
Student 1 : 1189195254
Grade_Sum : 0

Student 2 : 1603077354
Grade_Sum : 0

Student 3 : 974990924
Grade_Sum : 0

Student 4 : 1326149140
Grade_Sum : 0


lep@lep-Virtual-Machine:~/HW/FIle_Structure/HW2$
```

```
lep@lep-Virtual-Machine:~/HW/FIle_Structure/HW2$ cat Students.dat
201624548_Leejaeyun+201624548|lee jae yoon|Pusan|1252859056|62| 62925608|KIM|Seo
ul|694251221|79|&1326149140|BONG|Daejeon|1050682994|72|"411456048|Ann|Jeonju|487
304821|49|#1189195254|John|Daegu|898886005|23|$1346180012|Elen|Pohang|172230673|
78|"416087198|Yoon|Daegu|257061816|62|#1603077354|Choi|Ulsan|261501586|41|"97499
0924|Moon|Busan|341947808|92|"1128926133|Sun|Dokdo|289706544|31|lep@lep-Virtual-
Machine:~/HW/FIle_Structure/HW2$ cat CourseRegistration.dat
201624548_Leejaeyun@1508483348|2|1128926133|0|1603077354|0|416087198|0|134618001
2|0|>1090312074|3|974990924|0|1189195254|0|1128926133|0|62925608|0|K2100831715|1
|1189195254|0|1128926133|0|974990924|0|201624548|0|411456048|0|058458601|3|16030
77354|0|974990924|0|201624548|0|?331903174|3|1326149140|0|201624548|0|1346180012
|0|1189195254|0|@1491335883|3|1346180012|0|1189195254|0|1603077354|0|974990924|0
|L236021924|3|1326149140|0|1603077354|0|416087198|0|1128926133|0|1346180012|0|?1
932554477|3|411456048|0|1128926133|0|1346180012|0|201624548|0|%414968913|2|11289
26133|0|201624548|0|?367666014|3|1189195254|0|1603077354|0|974990924|0|132614914
0|0|lep@lep-Virtual-Machine:~/HW/FIle_Structure/HW2$
```