**POSCO BIGDATA ACADEMY 11기**

**성적 처리 프로그램**

**1Week - Assignment**

**담당교수: 윤은영**

**학번: B2**

**학과: 경영학과**

**이름: 윤민기**

**POVIS ID: s\_0716**

|  |
| --- |
| **명예서약(Honor code)**  **“나는 이 프로그래밍 과제를 다른 사람의 부적절한 도움 없이 완수하였습니다.”** |

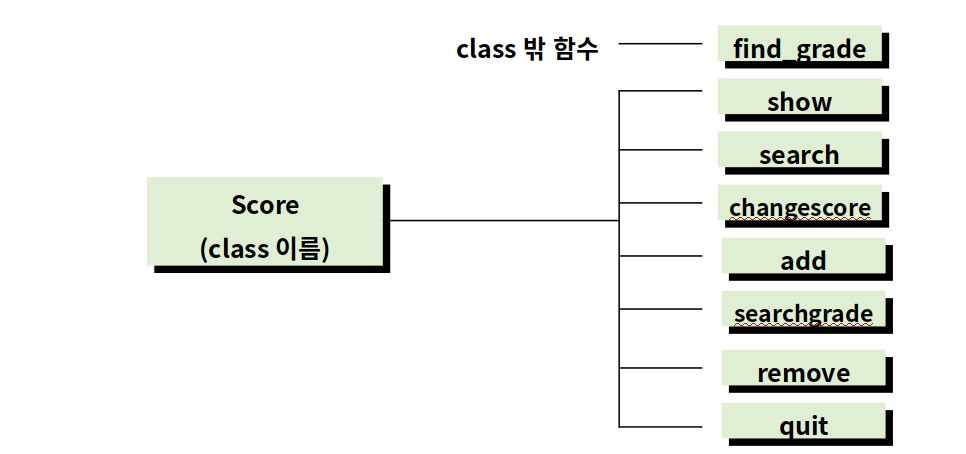
**Problem: 성적 처리 프로그램 개발**

1. 문제의 개요

본 프로그램을 간략히 설명하면 다음과 같다.

* terminal 환경에서 project.py 를 students.txt 파일을 읽어 실행시킨다.
* show, search, chagescore, add, searchgrade, remove, quit 총 7가지의 성적 출력 및 조작 기능을 구현한다.
* 사용자의 선택에 따라 조작된 성적 처리 파일을 newStudents.txt file로 저장한다.

이 때 사용되는 구상 가능한 구조 차트(structure chart)는 아래와 같이 표현될 수 있다.



* 입력부: 사용자가 입력하는 5개의 정수를 적절한 변수에 저장한다.
* 처리부: 변수에 저장된 값을 사용하여 점-직선 사이의 거리를 계산한다.
* 출력부: 계산이 완료된 거리 결과를 실수형태로 출력한다.

출력부: 자신의 이름을 ASCII 코드를 통해 출력한다.

이 외에 class 내에서 데이터 선언, 저장을 위한 \_\_init\_\_() 함수와 class 전체 실행을 위한 main() 함수가 있다. 위 차트에서는 class의 기능적 요소 파악에 용이하도록 두 함수를 생략했다.

2. 알고리즘

본 프로그램 작성을 위한 알고리즘을 Pseudo 코드 형태로 나타내면 다음과 같다.

|  |  |
| --- | --- |
| **Pseudo-algorithm for dot-line distance** | |
| 1  2  3  4  5  6  7 | *// 프로그램에 필요한 변수들은 미리 선언해놓은 것으로 가정한다.*  include essential headers <stdio.h> & <math.h>  prompt user and read coordinate of point <x, y>  prompt user and read coefficients of linear equation (a, b, c)  calculate the distance  Numerator = abs(ax+by+c)  Denominator = sqrt(a2+b2)  Distance = Numerator/Denominator  return Distance  print UserName using ASCII code |

class Score 내의 모든 함수는 상호 독립적이다. 각 함수의 실행이 데이터 값에는 변화를 줄 수 있지만 함수 서로 간에 영향, 선행 조건 등은 없다.

프로그램 알고리즘 설계에서 가장 먼저 신경을 쓴 부분은 데이터의 저장 여부이다. quit 명령어 입력 전까지 무한히 명령어 입력이 가능한 프로그램인 만큼 저장 부분에서 에러가 날 가능성이 높다고 생각했다.



3. 프로그램 구조 및 설명

a) 점의 좌표 및 직선의 계수 입력

* 프로그램을 실행하면 점의 좌표 <x, y>를 입력할 것을 요청한다. 사용자가 공백으로 구분하여 각 좌표를 입력하면 scanf를 통해 정수형 변수 x, y에 저장된다.
* 이 후, 위와 유사하게 직선의 계수 (a, b, c)를 입력할 것을 요청하며, 같은 방식으로 scanf를 통해 정수형 변수 a, b, c에 각각 저장된다.

b) 점과 직선 사이의 거리 계산

* 입력을 마친 후 각 변수에 저장된 값을 통해 점과 직선 사이의 거리를 계산하게 된다. 이 때, 점과 직선 사이의 거리 공식을 바탕으로 분자(numerator)와 분모(denominator)로 구분하여 계산한다.
* 분자는 math 헤더에 포함된 절대값 함수 abs()를 활용하여, 아래의 결과를 저장한다.
* 분모는 math 헤더에 포함된 제곱근 함수 sqrt()를 활용하여, 아래의 결과를 저장한다.
* 마지막으로 분자와 분모를 나누어 거리 결과에 저장한다.
* 결과를 출력할 때, “%.2f” 출력포맷을 사용하여 소수점 아래 둘째 자리까지 표현되도록 한다.

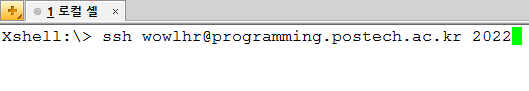
c) ASCII 코드를 통한 이름 출력

* 아래 소스 코드를 컴파일하고 실행하면, 각 영문에 해당하는 ASCII 코드를 알 수 있다.

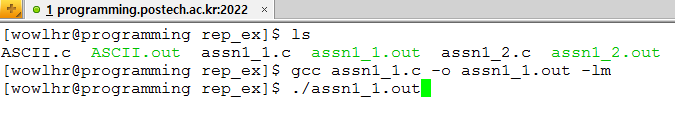
|  |  |
| --- | --- |
| **Source-code for ASCII code of input** | |
|  | #include <stdio.h>  int main(){  char init;  printf(“Input a character: “);  scanf(“%c”, &init);  printf(“\nASCII code of your input: %d\n”, init);  return 0;  } |

* 또는 ASCII code table을 참조하여 각 문자에 해당하는 ASCII 코드를 찾을 수 있다.
* printf(“%c%c%c …”, .ASCII\_1, ASCII\_2, …); 포맷을 활용하여 출력한다.

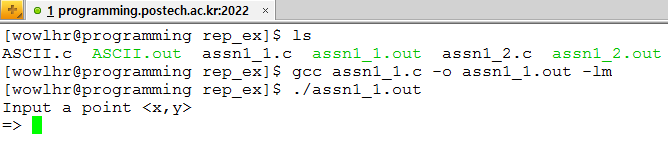
4. 프로그램 실행방법 및 예제



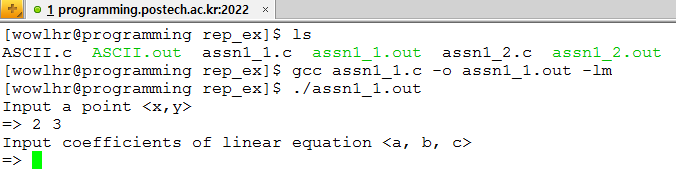
• Xshell 환경에서 위 명령어를 통해 비밀번호를 입력하고, 본인에게 할당된 리눅스 서버에 접속한다. 이 후 본 프로그램이 저장되어 있는 폴더로 이동한다.



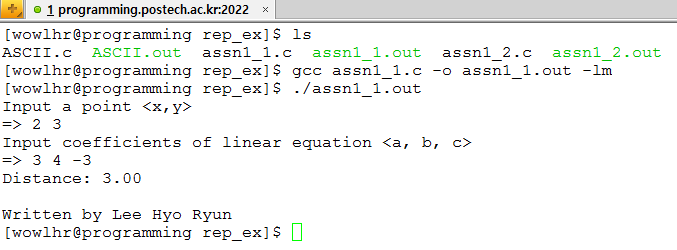
• 이동한 폴더에서 작성한 프로그램의 소스코드가 있는지 확인을 한 후 컴파일을 수행한다. 이 후, 컴파일이 완료된 파일을 실행시킨다.



• 위와 같이 점의 좌표를 요구하게 되고, 각 좌표를 공백으로 구분하여 입력을 완료한다.



• 점의 좌표 입력이 완료된 후, 직선 형성에 필요한 계수들을 요구한다. 점의 좌표 입력과 같은 방식으로 각 계수들을 입력해준다.



• 제시된 문제와 동일한 입력을 완료하였을 경우, 그 결과가 성공적으로 계산되어 실행예제와 동일한 결과를 출력하게 되었다.

• 또한, 프로그램이 종료되기 직전 ASCII 코드를 통한 이름이 정확히 출력됨을 확인할 수 있다.

5. 토론

* 문자의 ASCII 코드를 알아내는 프로그램을 작성할 때, scanf를 연속적으로 사용할 경우 정상적으로 동작하지 않는 문제가 발생하였다.
* 이는 문자를 입력 받을 때 사용한 ‘엔터’가 입력 버퍼에 남아있기 때문에 발생함을 알게 되었다.
* 문제를 해결하기 위해서는 ASCII 코드를 알아내는 프로그램을 필요한 만큼 반복 실행하거나, flushing을 통해 입력버퍼를 비워주는 작업(fflush(stdin))을 매 scanf 앞에서 행하는 방식이 요구된다.

6. 결론

* 본 과제에서는 prinf와 scanf를 사용하는 방법을 익히는데 유용했으며, ASCII 코드를 사용하는 과정을 통해 문자 및 정수 변수의 관계를 학습할 수 있게 되었다.

7. 개선방향

* 처음 프로그램을 설계할 때 코드 전반에 걸쳐 사용할 자료형으로 list() 가 편리해 보여 채택하였다. 그러나 실제로 코드를 구현한 결과 student\_id 라는 key 값으로 호출이 필요한 경우가 많았으며 list() 는 이 때 효율적인 도구가 아니다.

당장의 편리함에 눈이 멀어 list() 를 선택한 결과 나무를 보지 못한 결과를 낳았다. 다음에 비슷한 유형의 프로그램을 설계하게 된다면 dict() 를 우선순위에 둔다.