UserInfoList를 사용하려면 먼저 userinfolist.hpp 헤더를 프로젝트에 추가합니다.

이 헤더에는 사용자가 저장한 파일하나의 정보를 저장하는 UserData클래스, 사용자의 정보를 저장하는 User클래스, 사용자들을 저장하는 UserInfoListClass로 구성되어 있습니다.

이 UserInfoListClass 를 인스턴스로 생성해서 UserInfoList로 사용하시면 됩니다.

다음은 UserInfoList를 전역으로 생성하고 사용하는 예제의 설명입니다.

```
42  int main()
43  {
44
45      GenKeyPair();
46
47      UserInfoList.AddUser("a");
48      User& u = UserInfoList.FindUser("a");
49
50
51      u.AddUserData("DATA1", UserData(1, 1, "127.0.0.1:8080", "127.0.0.1:8080"));
52
53
54      u = UserInfoList.FindUser("a");
55      UserData& b = u.FindData("DATA1");
56      cout << b.BackUpFileAddress << b.BlockIndex << b.FileAddress << b.HashIndex <<  endl;
57
58      cout << u.UserName << u.MiningCount << u.UsageCount <<  endl;
59      u.MiningSuccessHandle();
60      cout << u.UserName << u.MiningCount << u.UsageCount << endl;
61
62      return 0;
63  }
64
```

45 : 이 함수는 RSA공개키(랜덤문자열)을 생성하는 함수입니다. 당장에 RSA인증을 구현할 계획은 없지만 추후를 위해서 이 함수에서 생성된 RSA공개키값을 사용자 이름으로 사용하려고 추가했습니다.

47 : AddUser 매서드는 UserInfoList에 새로운 사용자를 추가하는 매서드입니다. 만약 첫번째 인자로 전달된 사용자가 이미 존재한다면 FALSE를 리턴 하고 성공하면 TRUE를 리턴합니다.

48 : FindUser 매서드는 첫번째 인자로 전달된 사용자에 대한 User 클래스를 리턴받는 함수입니다. 이 매서드를 통해 특정 사용자를 찾을 수 있습니다.

51: AddUserData는 User클래스에서 새로운 파일정보를 추가할 때 사용하는 매서드입니다. 첫번째

인자로 추가할 파일정보의 이름, 두번째 인자로 UserData클래스가 전달됩니다.(생성자 순서는 해당 파일의 블록인덱스, 블록안에서의 해쉬 인덱스, 원본파일을 가지는 노드 주소, 백업파일을 가지는 노드 주소 입니다.) 첫번째 인자로 전달된 파일정보의 이름이 이미 존재한다면 false를 리턴합니다.

55 : FindData 매서드는 User클래스에서 파일정보의 이름을 가지고 UserData 클래스를 가져올 때 사용하는 매서드입니다 .

56 : 가져온 UserData 의 필드에 접근하는 모습입니다.

59: 노드에서 해당 User가 채굴에 성공했을 때 호출되는 함수입니다. 현제 구현은 업로드할 수 있는 파일 개수를 5개 늘리는 것으로 구현되어 있습니다. (디폴트 값도 5)