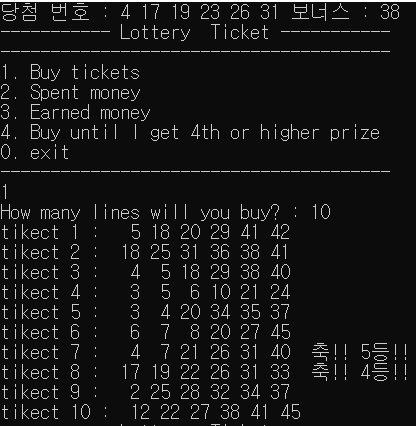
**소프트웨어 입문 설계 여섯 번째 과제**

|  |  |
| --- | --- |
|  |  |
| 1번을 통해 10개씩 두번 구매 -> 2번과 3번을 통해 구매한 티켓 수와 금액, 당첨 금액 등 확인 -> 4번을 통해 4등 이상이 나올 때까지 구매 -> 1번을 통해 5개 구매 -> 2번과 3번을 통해 지금까지 당첨상황 확인 -> 1,2,3,4,0 빼고 나머지 값 입력 시 예외 처리 -> 0번 입력 시 탈출 | |

2019042497 송정명

이차원 배열의 동적 할당을 하기 위해 일차원 배열이 담길 이차원 공간을 먼저 동적 할당하고, 그 공간에 각 일차원 배열들을 모두 동적 할당했습니다. 또한, 처음 실행할 때는 malloc() 함수를 사용했고, 추가로 구입할 때는 realloc() 함수를 사용해 공간을 늘리는 방식을 사용했습니다.

종료하기 전에 동적할당으로 사용했던 메모리를 해제하기 위해 먼저 일차원 배열이 할당되었던 공간의 메모리를 해제한 후, 그 일차원 배열의 포인터를 저장했던 이차원 공간들의 메모리를 해제했습니다.

총 구매한 티켓의 개수를 표현하기 위해 ticket이라는 int형 변수를 선언했고, 당첨횟수를 표현하기 위해 win[5]라는 int형 배열을 선언해 win[i+1]에 i등한 횟수를 담도록 했고, 총 당첨 금액을 표현하기 위해 money라는 int형 변수를 선언했습니다. 또한, 메뉴의 4번에서는 독립적으로 실행되어 출력해야 하기 때문에 4등 이상이 당첨되었음을 나타내는 win1, 4번이 실행되는 동안 구매한 티켓의 수인 ticket1, 4번이 실행되는 동안 번 돈인 money1의 int형 변수를 추가로 선언해주었습니다.

과제 조건으로 주어진 void printMenu()와 int\* newTicket() 함수 외에 추가로 int winTicket(int ans[], int\* com)을 만들어서 정답 티켓의 번호와 구매한 티켓의 번호가 같은 지 비교하고 등수를 출력해주는 함수를 만들었습니다.

추가로 1등이 여러 번 나오면 int 범위를 넘어버리지만 백만 번 돌려본 결과 1등이 한 번도 나오지 않은 것으로 보아 그럴 확률이 희박하다고 생각해 longlong타입을 쓰지 않았습니다.