비주얼 스튜디오 디버깅 사용법  
중단점 창: 디버그-창-중단점 하면 중단점에대한 목록이 나옴. 일괄적으로 지우거나, 필요하긴 한데 이번에는 안걸리게 하고싶으면 그럴 때 사용하면 편리하다.

**반복문 안쪽에서 어떤 조건이 만족했을때만 break를 걸고 싶을 때.** 그냥 if문으로 해서 break 걸어도 되지만 코드가 변경이 되는 상태니까 빌드를 같은 소스로 하더라도 처음 빌드랑 나중에 지운 빌드랑 다른 빌드로 쳐져서 문제가 될 수 있음 서버에서는. 그래서 **break포인트에서 우클릭해서 “적중횟수” 에 들어가서 몇번 돌아갔을 때 중단됐냐 할 수 있다.** 그럼 실제로 break가 50번 걸렸을때 비로소 break가 걸리게 될거임.  
또 다른 방법은 **우클릭-“중단점 조건”**에 들어가서 조건문을 넣을수있다. Icnt == 50 이렇게 넣으면 50번째에 멈추게 됨.  
또 또다른 방법은 break나 print걸지 않고 지금 로직이 돌 때 변수의 값을 실시간으로 눈으로 보고싶다면? 그럼 break포인트 우클릭 – “적중될 때” 누르면 됨. 그럼 메시지 표시가 있고 그 안에 묘한 내용이 있다. 그럼 중괄호 넣고 변수이름을 넣으면 그 변수가 출력이 된다. COUNT의 값은 : {icnt} 이렇게 넣으면 출력창에 출력이 된다.그럼 break포인트가 다이아몬드 모양으로 바뀜. 그리고 실행하면 출력창에 나오게 된다 실시간으로. 근데 단점은 어마어마하게 느리다. 이걸 실제 코드에 쓰면 프레임이 엄청 떨어져서 막 쓸 대상은 아니다. 나중에 멀티스레드 할때도 콘솔에 printf나 파일출력도 제대로 못한다. 로그 출력도 느려서 제대로 못한다. 그래서사실은 이런건 간단한 문제해결로 쓰이는거지 나머지는 고차원적으로 해야한다. 그때는 메모리 버퍼에 로깅해서 추적해야한다. 보통그렇게한다.

조사식 같은 경우엔 내가 원하는 변수들의 정보가 나오는건데, 우클릭에 16진수표시하면 16진수로 나옴. 그리고 여기서 casting도 가능하다. 이래저래 디버깅하다보니 내 손에 있는게 주소값밖에 없다? 이 주소가 누군지도 모르는데 주소값만 손에 남았다면? 메모리 주소로서 이 값을 보고싶다면? 이걸 주소값으로서 실제값을 보고싶다면 \*(int\*) 로 앞에 붙여서 캐스팅하면 그 메모리 번지에서 값을 뽑아오게 되는 것. 그러면 진짜 변수처럼 사용가능하다. 이 기능은 많이 쓰게 될거다. 현재 내가 바라보는 소스코드 기준으로 나오니까. 무슨 뜻이냐? 내 지역이거나 전역인것만 조사식에서 확인 가능하다. 현재 위치에서 복수있는거. 내 스택도 아닌 것은 못본다. 다른 스택에 있는 또는 접근못하는 메모리(존재한다느 ㄴ가정하에) 보고싶다면 이런 방법을 통해서 다른 스택에 있는 것도 값을 추적할 수 있다. 배열같은경우 포인터로 파라미터 넘기면 하나하나 노가다로 arr[0] 이렇게 추적하는방법도 있지만, 어떤 포인터든 이름하고 , 개수 이렇게 적으면 그걸 뭐든지 배열로 바꿔서 보여준다. Arr, 100 하면 100개짜리 배열로 보여준다. 원래 배열 크기보다 숫자를 크게 적으면 실제 무슨값이진 모르지만 그냥 배열형태로 보여주는 것 뿐이다.

메모리 뷰에서도 값을 바꿔도 된다. 그럼 그냥 데이터가 바뀜.

그리고 ip레지스터 설명할 때 보여주긴 했는데 오른클릭하고 나의 앞지점이나 다음지점으로 실행코드를 옮기는것도 가능하다. 근데 이건 롤백이 아니라 그냥 ip레지스터를 바꾸는거다. 권장할 내용이 아니다. 그냥 단순한 반복코드인데 껐다키긴 그렇고 한번더 돌려보고싶은 경우에 이런식으로 해서 변수 수동으로 초기화해서 돌려볼 수 있다. 강제적으로 위치 이동 시키는 방법. 쓰기가 아닌 체크, read정도만 하는 코드라면.

스레드는 나중에

그리고 또 한가지 쓸모 있기도 없기도 한 내용인데   
메모리의 변경을 탐지해주는 break 포인트. 성능은 당연히 떨어지고. 미리잡을수없다. 일단은 실행시켜서 디버그모드로 중단점 걸린 상태로 정지가 되어야 한다. 그다음에 중단점창에서 새로만들기에서 새 데이터 중단점을 누른다. 여재까지느 다 함수에서 중단점이다. 데이터 중단점이라는게 있다. 그럼 여기에 주소를 적는다. 정말로 메모리 번지를 적거나 변수이름을 &하면 알아서 주소로 변환해준다. 그러면 그 메모리에 변경이 있는 지점이 나오면 창을 띄워주고 알려준다. 근데이것도 변수를 알고있을때의 이야기.

또 가끔가다가 무한루프도는거 만들면 그냥 실행하다가 이상하면 그상황 바로 디버깅하고싶으면 끄는거 아니고 그냥 돌고있을 때 프로세스를 잡는다 디버깅으로. 돌고있는상태에서 디버깅ㅇ하고싶다? 디버그-프로세스에 연결-그창에서 내 실행파일 이름을 클릭하고 연결을 누른다.그럼 그 상태로 디버깅 모드로 바뀐다. 프로세스를 잡았기떄문에. 전제는 내가만든 소스랑, 내가만든 pbd? Pbb? 가 그대로 있다는 전제. 대부분 그냥 하면 된다. 프로세스 연결은 원격 디버깅도 된다. 원격에 있는 컴퓨터를 대상으로도 들어갈수있따. 대신 상대쪽에서 그 프로그램을 구동해야하니까. 가끔 재현도 안되고 기회가 아까운 상황, 증상이 나타났는데 지금이아니면 못보는 1달에 한번 나오는 버그 그런것들 서버끌마음으로 원격디버깅 하기도 한다. 라이브서버에. 거기서 break걸면 유저들은 다 튕김. 그럼에도 좋은기회라 잡는 것. 다른 프로세스도 접근간으하지만 지금 코드가 없으니까 디스어셈블러로 나올테지만.

화면의 깜빡임 없애기.  
콘솔이든 그래픽이든 gdi든 방법은 더블버퍼링이라는 방법이다. 모니터가 있을 때 화면에 그리고 다음프레임을위해 지우고 변경된 그립을 그리고 지우고 그러는게 지우는것 때문에 깜빡인다. 지우는걸 안보이게하려면 화면크기랑 같은 버퍼를 하나 더만들고, 거기에 그려둠. 그리고 그걸 그대로 화면에 덮어써버리는 것. 그 다음에 프레임을 그릴거라면 다른 화면에 지우고 그리고 덮어쓰고. 최종화면은 지우지 않는것이다. 덮어쓰기만 하는 것. 그래서이걸 더블버퍼링이라고 한다.

char형태의 버퍼를 잡는다. 화면크기랑 똑 같은 크기의. [81][24]짜리의. 그럼 이제 우리는 그리는건 뭐든지 여기다가 할 거임. 그리고 이 버퍼를 통으로 줄단위로 출력할거임. 80개의 글자가 24줄인것. 한줄한줄 원래는 printf로 출력하려는게 목적. Movecursur는 첫 줄로 커서를 옮겨주는것밖에 안할거임. 땡기고 출력을해야 덮어쓰는 개념이니까.

지점 잡아서 printf 반복문 24번 하자는 이야기.

이걸 flip이라고 부를거다. 원래 더블버퍼링은 비디오메모리입장에서는 어떤개념이었냐? 실제 모니터가 있으면 여기 내부의 비디오메모리에 모니터는 비디오메모리를 읽어다가 한줄씩 주사하는게 일이었다. 비디오메모리의 내용을 비디오카드가 한줄씩 주사하는 것. 근데 원래 개념은 주사를 버퍼 2개를 번갈아가면서 한다는 뜻이다. 플립이라는건 뒤집는 것. 그런개념이라 더블버퍼링에서 화면에 보이게끔한다는걸 플립이라고 말한다. 실제그게아니어도 개념은그거니까. 문제는 횡스크롤같은 게임에서 다음프레임에 화면이 너무 많이 바뀌면 티어링이라고 화면이 찢어지는것처럼 보인다. 보통게임은 싱크를 안맞춘다.

여기서도 플립이라고 표현하겠지만 우리는그냥 화면출력의 의미.

Bufferclear는 버퍼지우기. 2차원 char배열을 초기화. 그 초기화는 space로 채운다. 널로 채우는게 아니라

그리고 sprite 호마녀에 띄워질 그림이나 사진을 스프라이트. Sprite\_draw는 좌표가들어오면 특정캐릭터를 버퍼에 찍는거다. 버퍼상의 좌표 위치에 캐릭터를 대입해주면 된다.

순서는  
1. 스크린버퍼 지우기  
2. 스프라이트 출력  
3. 스크린버퍼 화면으로 출력

Main  
1.초기화  
2.while  
3.키보드 입력() getasynckEYSTATE. 보통 입력과 로직을 분리하려면 이벤트 큐에다가 키 입력을 넣어야 됨.  
4.로직()  
5.랜더링()  
 플레이어를 버퍼에 랜더링 / 적군 버퍼에 랜더링 / Flip()