재귀함수

재귀는 트리형 구조로 레벨에 따른 이동을 파악해야한다.

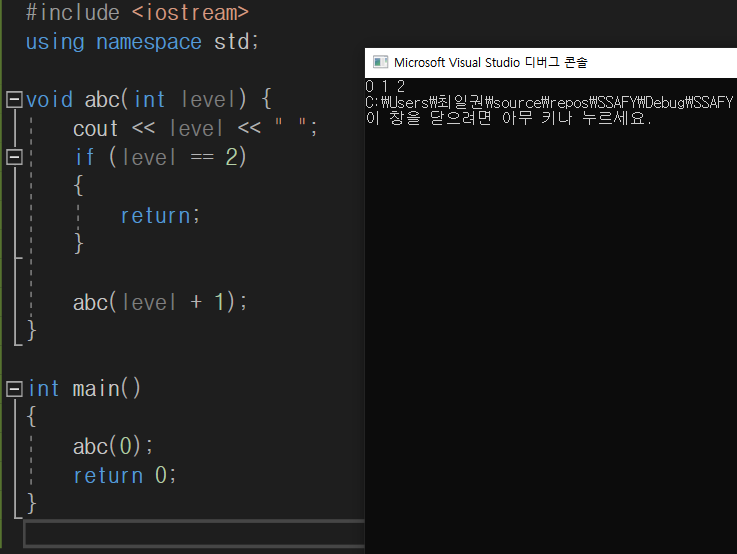
#include <iostream>  
using namespace std;  
​  
void abc(int level){  
     
   if(level==2)  
  {  
       return;  
  }  
     
   abc(level+1);  
}  
​  
int main()  
{  
   abc(0);  
   return 0;  
}

위의 경우 level이 0일 경우부터 시작해 2에서 멈춰 return을 만나 함수가 종료된다.

만약 level을 출력한다면

#include <iostream>  
using namespace std;  
​  
void abc(int level){  
   cout<<level<<" ";  
   if(level==2)  
  {  
       return;  
  }  
     
   abc(level+1);  
}  
​  
int main()  
{  
   abc(0);  
   return 0;  
}

위의 출력 값은 0 1 2로 종료될 것이다.

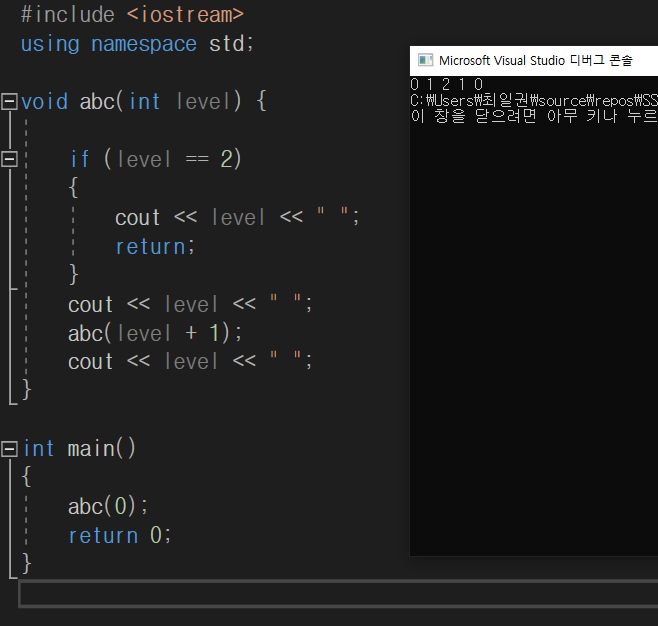


level은 재귀를 돌고난 후는 원래의 값을 유지하게 되는데

01210을 출력 값으로 기대하고 함수를 짜게 된다면

#include <iostream>  
using namespace std;  
​  
void abc(int level){  
     
   if(level==2)  
  {  
       cout<<level<<" ";  
       return;  
  }  
   cout<<level<<" ";  
   abc(level+1);  
   cout<<level<<" ";  
}  
​  
int main()  
{  
   abc(0);  
   return 0;  
}

위와 같은 구조로 함수를 짠다면 출력 값으로 01210이 도출될 것이다.

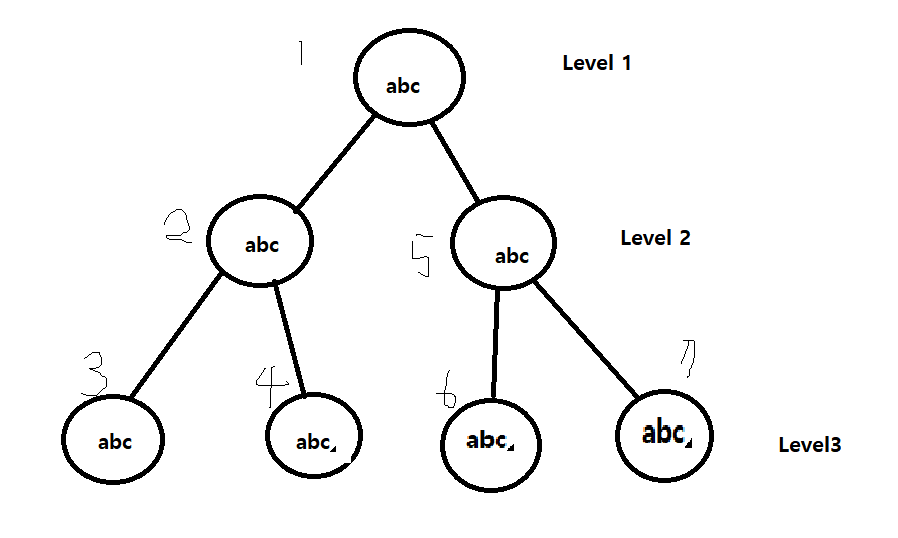


재귀라는 것은 결국 하나의 트리 형태로 이루어 지기 때문에 재귀를 짠다면 어떤 트리형으로

이루어져있는지를 파악하는 것이 중요하다.

#include <iostream>  
using namespace std;  
​  
void abc(int level){  
     
   if(level==2)  
  {  
       return;  
  }  
     
   for(int i=0;i<2;i++)  
  {  
       abc(level+1);  
  }  
}  
​  
int main()  
{  
   abc(0);  
   return 0;  
}

위 구조를 level를 trace해보면 정확하게 파악할 수 있다.



위 함수의 구조를 tree형태로 표현한 것

메인에서 abc()함수를 돌고난 후 for문에서 차례대로 돌게 될것인데 이때 함수 안에 함수를 시행하게 된다면

return까지 돌고난 후 그 전 함수의 원래 위치로 되돌아가게 된다.

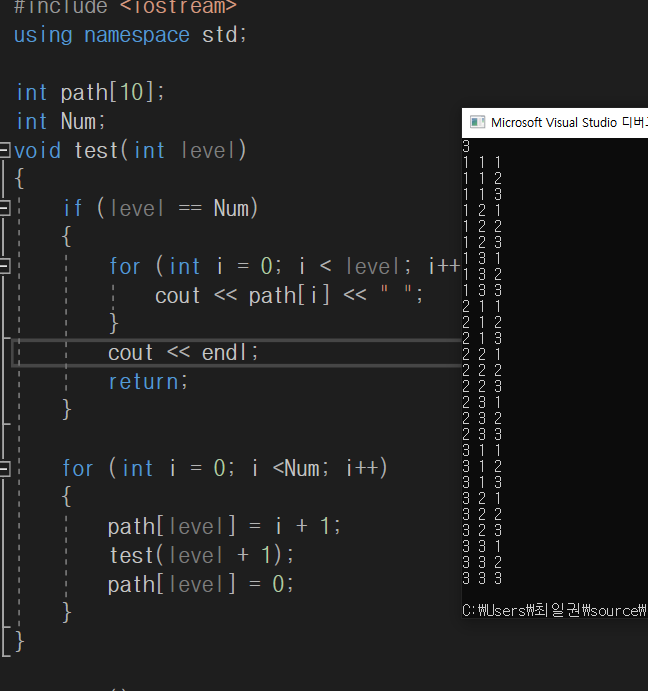
재귀를 이용한 순열

#include <iostream>  
using namespace std;  
​  
int path[10];  
int Num;  
void test(int level)  
{  
   if(level==Num)  
  {  
       for(int i=0;i<level;i++){  
      cout<<path[i]<<" ";  
      }

cout<<endl;  
       return;  
  }  
     
   for(int i=0;i<Num;i++)  
  {  
       path[level]=i+1;  
       test(level+1);  
       path[level]=0;  
  }  
}  
​  
int main()  
{  
   cin>>Num;  
   test(0);  
     
   return 0;  
}

1차원 배열 path에 수를 저장한 후 재귀를 돌린다음 level이 원하는 num에 도달하게 된다면

1부터 시작한 수를 num까지 출력할 수 있다.(돌려보면 바로 앎)



하지만 위의 함수는 중복이 허용된 함수이기 때문에 중보를 허용하지 않으려면

Select라는 배열을 추가해 추가되었는지를 확인하는 과정이 필요하다.

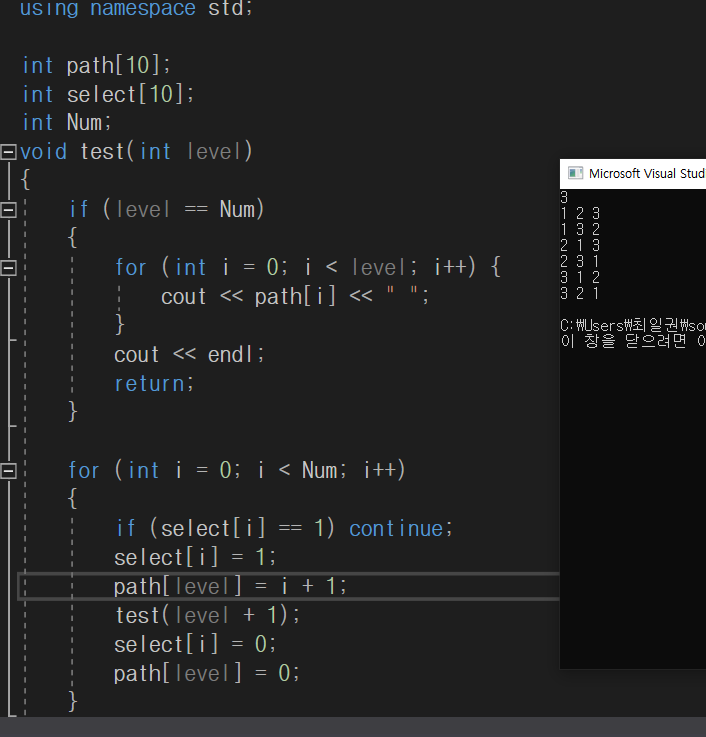
#include <iostream>  
using namespace std;  
​  
int path[10];  
int select[10];  
int Num;  
void test(int level)  
{  
   if(level==Num)  
  {  
       for(int i=0;i<level;i++){  
      cout<<path[i]<<" ";  
      }

cout<<endl;  
       return;  
  }  
     
   for(int i=0;i<Num;i++)  
  {  
       if(select[i]==1) continue;  
       select[i]=1;  
       path[level]=i+1;  
       test(level+1);  
       select[i]=0;  
       path[level]=0;  
  }  
}  
​  
int main()  
{  
   cin>>Num;  
   test(0);  
     
   return 0;  
}

위의 과정을 추가한다면 숫자가 들어온지를 파악하고 들어오지 않은 숫자만

path에 추가하기 때문에 1111과 같은 중복된 순열이 아닌 1234와 같이 서로 다른 숫자만

출력할 수 있게 한다.



pro.mincoding.co.kr에서 연습문제를 풀어봐~~

아이디: dlfrnjs3456

비밀번호: 일권3456