(훈련반2) Level29.5

DFS와 BFS를 한번 더 정리할 때가 왔습니다.

그래프 / 트리를 하드코딩 하는 방법은 세 가지 방법이 있습니다.

1. **인접행렬**
2. **인접리스트**
3. **1차원배열 (이진트리 전용)**

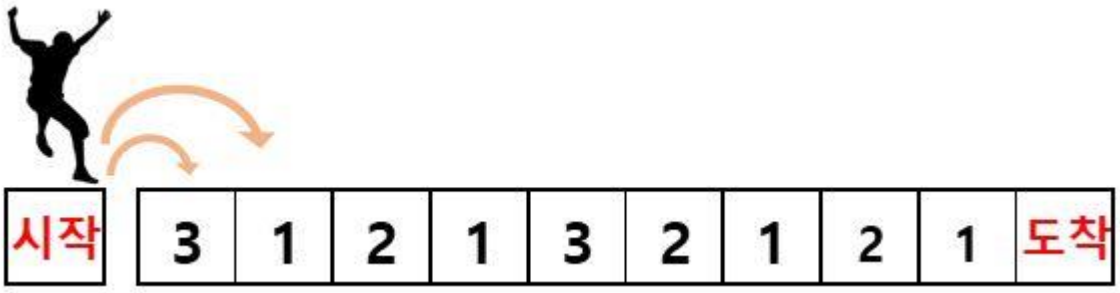
그리고 그래프 / 트리를 탐색하는 방법은 이렇게 두가지 방법이 있습니다.

1. **DFS (깊이우선탐색)**
2. **BFS (너비우선탐색)**

헷갈리지 않도록 유의 해 주세요.

Level29.5 징검다리 돌아오기

문제 1번 [숙제 목록보기]



처음 점프를 할 n값을 입력 받으세요.

만약,

1 을 입력 받으면 **3** 으로 점프하고,

2 를 입력 받으면 **1** 로 점프합니다.

그리고 **다음 점프는 바닥에 써 있는 칸만큼 점프를 계속 합니다.**

도착지점에 도달하면 return을 하게 되어 시작점으로 돌아옵니다.

이 과정을 모두 출력 해주세요.

(재귀호출로 구현 해 주세요)

ex)

<입력>

1

<출력>

시작 3 1 3 2 도착 2 3 1 3 시작

입력 예제

```
5
```

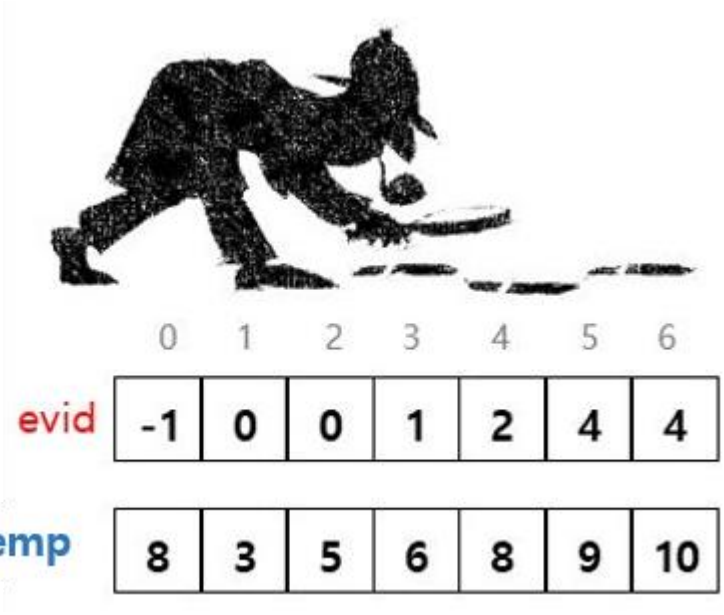
출력 결과

```
시작 3 2 도착 2 3 시작
```

```cpp
#include <iostream>
using namespace std;

const int arrSize = 11;
int arr[arrSize] = { 0,3,1,2,1,3,2,1,2,1,0 };
int path[arrSize] = {};
int pathIdx = 0;

void recursive(int _pos, int _jumpCnt)
{
	path[pathIdx++] = _pos;
	int nextJump = arr[_pos + _jumpCnt];
	int newPos = _pos + _jumpCnt;
	if (newPos != arrSize - 1)
		recursive(newPos, nextJump);
}

int main(void)
{
	int startPos = 0;
	cin >> arr[startPos];
	int jump = arr[startPos];
	recursive(startPos, jump);
	for (int i = 0; i < pathIdx; ++i)
	{
		int idx = path[i];
		int val = arr[idx];
		if (idx == 0)
			cout << "시작 ";
		else
			cout << val << " ";
	}
	cout << "도착 ";
	for (int i = pathIdx- 1; i >= 0; --i)
	{
		int idx = path[i];
		int val = arr[idx];
		if (idx == 0)
			cout << "시작 ";
		else
			cout << val << " ";
	}


	return 0;
}
```

Level29.5 범인의 흔적

문제 2번 [숙제 목록보기]



추적을 시작 할 index를 입력 받으세요.

만약 5를 입력 받았다면, **5번 index** 부터 추적을 시작합니다.

5번 index를 살펴보면 범인은 4번 index에서 출발했고, 9시에 도착했다는 것을 알 수 있습니다.

4번 index를 살펴보면 범인은 2번 index에서 출발했고, 8시에 도착했다는 것을 알 수 있습니다.

2번 index를 살펴보면 범인은 0번 index에서 출발했고, 5시에 도착했다는 것을 알 수 있습니다.

범죄자의 흔적들을 추적해가면, 마지막에는 -1에 도달합니다.

-1이 있는 곳에서 범죄자를 잡을 수 있습니다.

범인은 0번 index부터 몇 시에 몇 번 index로 이동했는지

순서대로 출력하세요.

(재귀를 이용해서 범인을 추적 해 주세요)

입력 예제

```
5
```

출력 결과

```
0번index(출발)
2번index(5시)
4번index(8시)
5번index(9시)
```

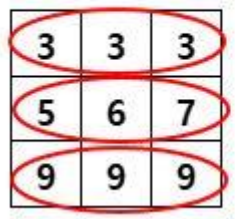
Level29.5 모두 같은 숫자일까? (난이도 : ★★☆)

문제 3번 [숙제 목록보기]

3x3 배열에 숫자를 입력해 채워줍니다.

그리고 가로로 한줄씩 모두 같은 숫자인지 검사하는 프로그램을 작성해주세요.

같으면 같은 숫자를 출력, 아니면 (소문자)x를 출력 하세요.



ex)

<입력>	<출력>
3 3 3	3
5 6 7	x
9 9 9	9

입력 예제

```
3 3 3
5 6 7
9 9 9
```

출력 결과

```
3
x
9
```

```cpp
#include <iostream>
using namespace std;

const int arrSize = 3;
int arr[arrSize][arrSize] = {};
const int offset = 48;   // ascii -> '0' = 48

int main(void)
{
	for (int y = 0; y < arrSize; ++y)
	{

		for (int x = 0; x < arrSize; ++x)
			cin >> arr[y][x];
	}
	char referArr[arrSize] = {};
	for (int x = 0; x < arrSize; ++x)
	{
		for (int y = 0; y < arrSize; ++y)
		{
			if(x == 0)
				referArr[y] = arr[y][x] + offset;
			else
			{
				if (referArr[y] != arr[y][x] + offset)
					referArr[y] = 'x';
			}
		}
	}

	for (int i = 0; i < arrSize; ++i)
		cout << referArr[i] << endl;

	return 0;
}
```

Level29.5 두 정렬되어있는 배열을 하나로 (난이도 : ★★★)

문제 4번 [숙제 목록보기]

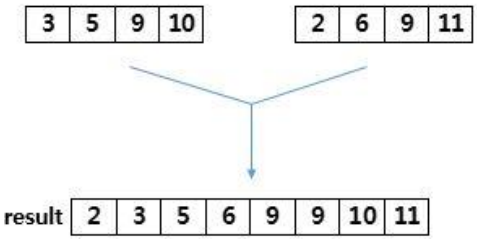
숫자 4개씩 2개의 배열에 숫자를 입력 받아주세요.

3	5	9	10

2	6	9	11

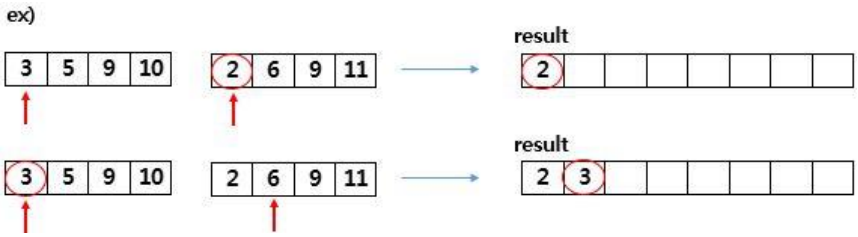
입력값은 정렬된 상태로 숫자가 들어옵니다.

이 두배열을 합쳐 정렬된 8개의 숫자를 저장 하려고 합니다.



이렇게 합치기 위한 알고리즘은

비교를 한 후에 작은 숫자를 result배열에 넣고 화살표를 옆으로 옮김니다.



위와 같은 동작을 반복하면, 정렬된 result 배열을 만들 수 있습니다.

위 알고리즘대로 코딩하여 result 배열을 만들고 출력 해주세요.

입력 예제

```
3 5 9 10
2 6 9 11
```

출력 결과

```
2 3 5 6 9 9 10 11
```

```cpp
#include <iostream>
using namespace std;

const int arrSize = 4;
int arr1[arrSize] = {};
int arr2[arrSize] = {};

int main(void)
{
	for (int i = 0; i < arrSize * 2; ++i)
	{
		if (i < 4)
			cin >> arr1[i];
		else
			cin >> arr2[i % 4];
	}
	int result[arrSize * 2] = {};
	for (int k = 0; k < arrSize; ++k)
	{
		int smallIdx = k * 2;
		int bigIdx = smallIdx + 1;
		if (arr1[k] < arr2[k])
		{
			result[smallIdx] = arr1[k];
			result[bigIdx] = arr2[k];
		}
		else
		{
			result[smallIdx] = arr2[k];
			result[bigIdx] = arr1[k];
		}
	}

	for (int o = 0; o < arrSize * 2; ++o)
		cout << result[o] << " ";

	return 0;
}
```

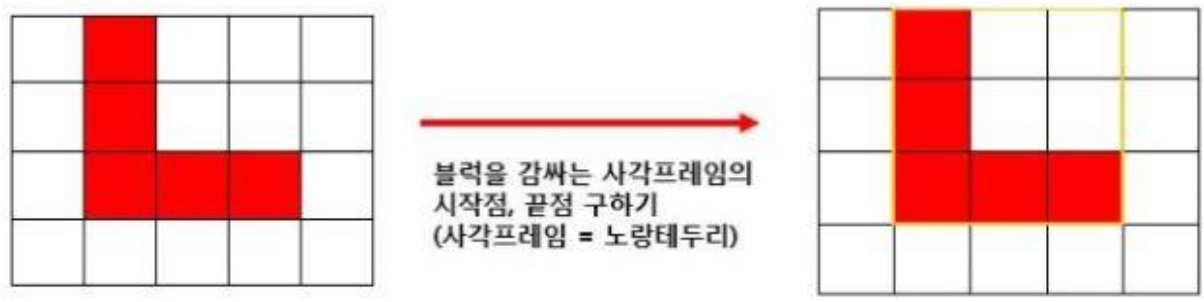
Level29.5 블럭을 감싸는 사각프레임 좌표 구하기 (난이도 : ★★★)

문제 5번 [숙제 목록보기]

4x5 2차배열이 있습니다.

블럭을 입력 받아주세요.

사각프레임에 블럭이 있을때, 이블럭을 감싸는 **사각프레임의 시작점과 끝점의 좌표**를 구해주세요.



ex1)

[입력]	[출력]
0 1 0 0 0	(0,1)
0 1 0 0 0	(2,3)
0 1 1 1 0	
0 0 0 0 0	

ex2)

[입력]	[출력]
0 0 0 0 0	(1,2)
0 0 1 1 0	(2,4)
0 0 1 1 1	
0 0 0 0 0	

입력 예제

```
0 1 0 0 0
0 1 0 0 0
0 1 1 1 0
0 0 0 0 0
```

출력 결과

```
(0,1)
(2,3)
```

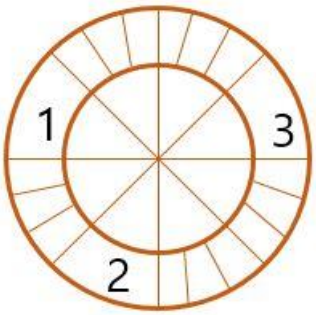
```cpp
#include <iostream>
using namespace std;

const int blockY = 4;
const int blockX = 5;
int block[blockY][blockX] = {};
const int exsit = 1;

int main(void)
{
	int blockCnt = blockY * blockX;
	for (int i = 0; i < blockCnt; ++i)
		cin >> block[i / blockX][i % blockX];

	int minX = blockX;
	int maxX = 0;
	int minY = blockY;
	int maxY = 0;

	for (int y = 0; y < blockY; ++y)
	{
		for (int x = 0; x < blockX; ++x)
		{
			if (block[y][x] == exsit)
			{
				if (y > maxY)
					maxY = y;
				if (y < minY)
					minY = y;
				if (x > maxX)
					maxX = x;
				if (x < minX)
					minX = x;
			}
		}
	}

	cout << "(" << minY << "," << minX << ")" << endl;
	cout << "(" << maxY << "," << maxX << ")";

	return 0;
}
```

Level29.5 톱니바퀴 (난이도 : ★★★)

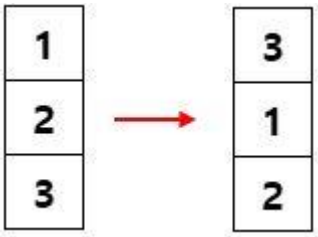
문제 6번 [숙제 목록보기]

3개의 숫자로 되어있는 톱니바퀴가 있습니다.

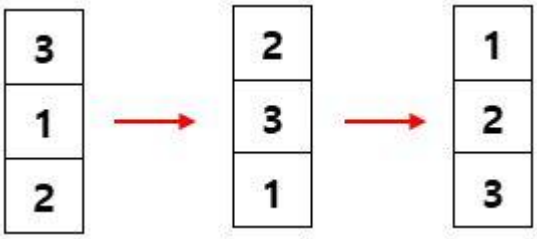
그림으로 그리면 아래와 같습니다.



이 상태에서 아래쪽으로 한칸 돌리면 아래와 같이 됩니다.



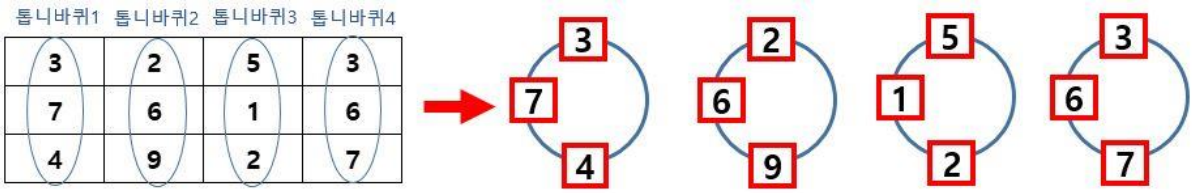
이 상태에서 두번 더 돌리면 아래와 같이 됩니다.



이런식으로 동작하는 톱니바퀴가 4개 있습니다.

각 톱니바퀴를 돌려 결과를 출력 하려고 합니다.

아래의 톱니바퀴 상태 배열을 하드코딩 해 주세요.



그리고 몇번 돌릴지에 대한 숫자 4개를 입력 받아주세요.

이 숫자 4개는 4개의 톱니바퀴를 각각 돌릴 횟수입니다.

숫재대로 톱니바퀴를 돌렸을 때 나온 결과를 출력 해주세요.

입력 예제

```
1 2 1 2
```

출력 결과

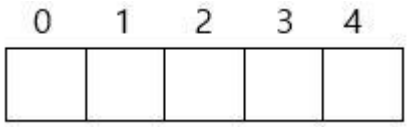
```
4626
3957
7213
```

```cpp
#include <iostream>
using namespace std;

const int wheelCnt = 4;
const int toothCnt = 3;
int wheel[toothCnt][wheelCnt] =
{
	3,2,5,3,
	7,6,1,6,
	4,9,2,7
};
int rotates[wheelCnt] = {};

void copy(int* _from, int _wheelNum)
{
	for (int i = 0; i < toothCnt; ++i)
	{
		wheel[i][_wheelNum] = _from[i];
	}
}

int main(void)
{
	for (int i = 0; i < wheelCnt; ++i)
		cin >> rotates[i];

	for (int m = 0; m < wheelCnt; ++m)
	{
		int tempWheel[toothCnt] = {};
		for (int n = 0; n < toothCnt; ++n)
		{
			int rotatCnt = rotates[m];
			int newIdx = (n + rotatCnt) % toothCnt;
			tempWheel[newIdx] = wheel[n][m];
		}
		copy(tempWheel, m);
	}

	for (int y = 0; y < toothCnt; ++y)
	{
		for (int x = 0; x < wheelCnt; ++x)
			cout << wheel[y][x];
		cout << endl;
	}

	return 0;
}
```

Level29.5 지렁이 놓기 (난이도 : ★★☆)

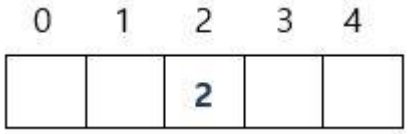
문제 7번 [숙제 목록보기]

다섯칸의 맵이 있고 지렁이를 올려두려고 합니다.



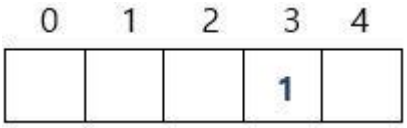
수명이 2인 지렁이를 2번 index에 올려두면 아래와 같이 됩니다.

배열안에 적은 값이 지렁이의 수명입니다



지렁이는 1초에 한번씩 오른쪽으로 한칸씩 이동합니다.

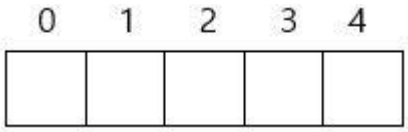
한칸씩 이동하면서 수명이 1씩 줄어 듭니다.



다음 1초 후에는 수명이 0이되어 지렁이는 죽게 됩니다.

만약 지렁이가 수명이 남았지만 맵 밖으로 나가도 죽게 됩니다.

(죽은 후에는 숫자 0을 표기하지 않습니다)



올려놓을 지렁이의 index와 **수명을 입력** 받고

지렁이가 죽을 때 까지 동작 결과를 출력하세요.

입력 예제

```
2 2
```

출력 결과

```
__2__
___1_
_____
```

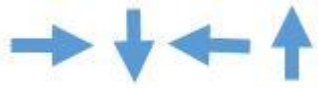
```cpp
#include <iostream>
using namespace std;
/*
	[문제]
	- 값이 곧 지렁이의 수명
	- 1초에 한칸씩 오른쪽으로 이동

*/
const int mapSize = 5;
int map[mapSize] = {};
int posIdx = 0;
int wormLife = 0;
const int emptyPlace = 0;
void print()
{
	for (int i = 0; i < mapSize; ++i)
	{
		if (map[i] == 0)
			cout << '_';
		else
			cout << map[i];
	}
	cout << endl;
}

int main(void)
{
	cin >> posIdx >> wormLife;
	map[posIdx] = wormLife;
	bool isAlive = true;
	while (isAlive == true)
	{
		print();
		map[posIdx] = emptyPlace;
		int newPos = ++posIdx;
		int leftLife = --wormLife;
		if (leftLife == 0 || newPos >= mapSize)
		{
			isAlive = false;
			print();
		}
		else
			map[newPos] = leftLife;
	}

	return 0;
}
```

Level29.5 움직이는 몬스터 (난이도 : ★★★)

문제 8번 [숙제 목록보기]

아래 배열에 게임 상태를 나타내는 MAP입니다.

색칠된 부분은 벽이고 **알파벳은 몬스터의 이름**입니다.

몬스터의 AI는 단순하여  만 순서대로 반복해서 움직입니다.

1초 후에는 오른쪽으로 움직이고,

2초 후에는 아래로 움직이고,

3초 후에는 왼쪽,

4초 후에는 위,

5초 후에는 다시 오른쪽으로 움직입니다.

이 몬스터는 벽을 통과하거나 몬스터끼리 겹치지 못합니다.

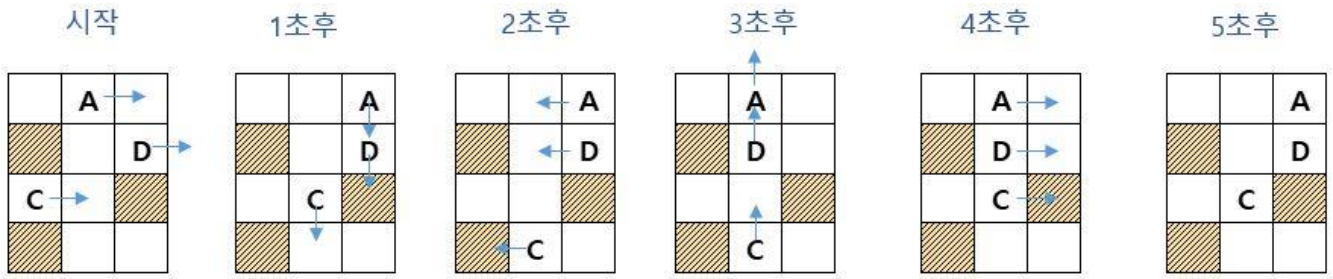
그래서 움직이려고 하는 곳이 막혀있다면 가만히 서있습니다.

*** 몬스터는 알파벳 순서대로 움직이게 됩니다.**

아래 예제에서는 A먼저 움직이고, C움직이고, D가 움직이게 됩니다.

MAP을 입력 받고, 5초 후 상황을 출력 해주세요.

***모든 몬스터는 1초에 한번씩 움직이게 됩니다.**



입력 예제

```
_A_
#_D
```

```
C_#
#__
```

출력 결과

```
__A
#_D
_C#
#__
```

```cpp
#include <iostream>
using namespace std;

const int directCnt = 4;
const int direction = 2;
int direct[directCnt][direction] =
{
        0,1,    // right
        1,0,    // bottom
        0,-1,   // left
        -1,0    // top
};
const int playTime = 5;

struct Tile
{
public:
        void setPos(int _y, int _x)
        {
                y = _y;
                x = _x;
        }
        void setState(char _state)
        {
                state = _state;
        }

public:
        Tile()
                : y(0)
                , x(0)
                , state('\0')
        {
        }
        int y;
        int x;
        char state;     // _ : empty , # : wall, A/B/C : user
};

const int mapY = 4;
const int mapX = 3;
Tile gameMap[mapY][mapX] = {};
const int userCnt = 3;
Tile users[userCnt] = {};
int usersIdx = 0;
```

```
void input()	// 초기 상태 입력받기
{
	for (int y = 0; y < mapY; ++y)
	{
		for (int x = 0; x < mapX; ++x)
		{
			gameMap[y][x].setPos(y, x);
			cin >> gameMap[y][x].state;
			// ascii -> 'A' = 65, 'Z' = 90
			if (gameMap[y][x].state >= 65 && gameMap[y][x].state <= 90)
				users[usersIdx++] = gameMap[y][x];
		}
	}
}

void move()
{
	int direction = playTime % 4;
	for (int i = 0;i < direction; ++i)
	{
		for (int k = 0; k < userCnt; ++k)
		{
			int newY = users[k].y + direct[i][0];
			int newX = users[k].x + direct[i][1];
			if (newY >= 0 && newY < mapY && newX >= 0 && newX < mapX)
			{
				char newPlace = gameMap[newY][newX].state;
				if (newPlace == '_')
				{
					gameMap[users[k].y][users[k].x].state = '_';
					gameMap[newY][newX].state = users[k].state;
					users[k].setPos(newY, newX);
				}
			}
		}
	}
}

void print()
{
	for (int y = 0; y < mapY; ++y)
	{
		for (int x = 0; x < mapX; ++x)
			cout << gameMap[y][x].state;
		cout << endl;
	}
}
```

```
int main(void)
{
        input();
        move();
        print();

        return 0;
}
```

Level29.5 같은단어 찾기 (난이도 : ★★★)

문제 9번 [숙제 목록보기]

두문장을 입력받으세요.**(최대15글자)**

그리고 **가장 긴 같은 단어**를 찾아주세요.

매우 어렵습니다

입력 예제

```
ABABCGKABABC
BTBCKABABCT
```

출력 결과

```
KABABC
```

```
#include <iostream>
using namespace std;

const int maxSize = 15;
char str1[maxSize + 1] = {};
char str2[maxSize + 1] = {};
char sameStr[maxSize] = {};
int sameLength = 0;

void compare()
{
	int strLen1 = strlen(str1);
	int strLen2 = strlen(str2);
	for (int i = 0; i < strLen1; ++i)
	{
		for (int k = 0; k < strLen2; ++k)
		{
			if (str1[i] == str2[k])
			{
				char temp[maxSize] = {};
				int tempLength = 0;
				int offset = 0;
				while (true)
				{
					int newIdx1 = i + offset;
					int newIdx2 = k + offset;
					++offset;
					if (newIdx1 < strLen1 && newIdx2 < strLen2)
					{
						char ch1 = str1[newIdx1];
						char ch2 = str2[newIdx2];
						if (ch1 == ch2)
						{
							temp[tempLength++] = ch1;
						}
						else
							break;
					}
					else
						break;
				}
```

```
                                if (tempLength > sameLength)
                                {
                                        sameLength = tempLength;
                                        for (int i = 0; i < strlen(temp); ++i)
                                                sameStr[i] = temp[i];
                                }
                        }
                }
        }
}

int main(void)
{
        cin >> str1 >> str2;
        compare();
        cout << sameStr;

        return 0;
}
```