동국대학교
dongguk university

제어문 - 반복문

HyoJoon Han
동국대학교
han6343@dongguk.edu

목차

반복문과 배열

- 기본적인 배열 생성 방법

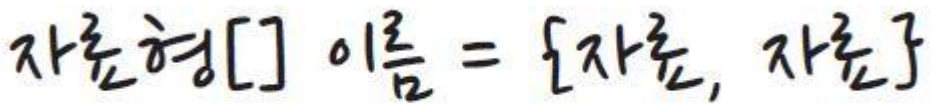


그림 4-1 배열 선언 형식

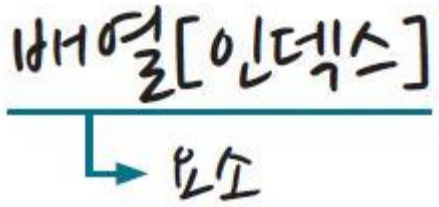


그림 4-2 배열의 요소

코드 4-3 기본적인 배열 생성 방법 /4장/Arrays

```
static void Main(string[] args)
{
    int[] intArray = { 52, 273, 32, 65, 103 };
    long[] longArray = { 52, 273, 32, 65, 103 };
    float[] floatArray = { 1.0F, 2.0F, 3.0F, 4.0F, 5.0F };
    double[] doubleArray = { 1.0, 2.0, 3.0, 4.0, 5.0 };
    char[] charArray = { '가', '나', '다', '라' };
    string[] stringArray = { "윤인성", "연하진", "윤아린" };
}
```

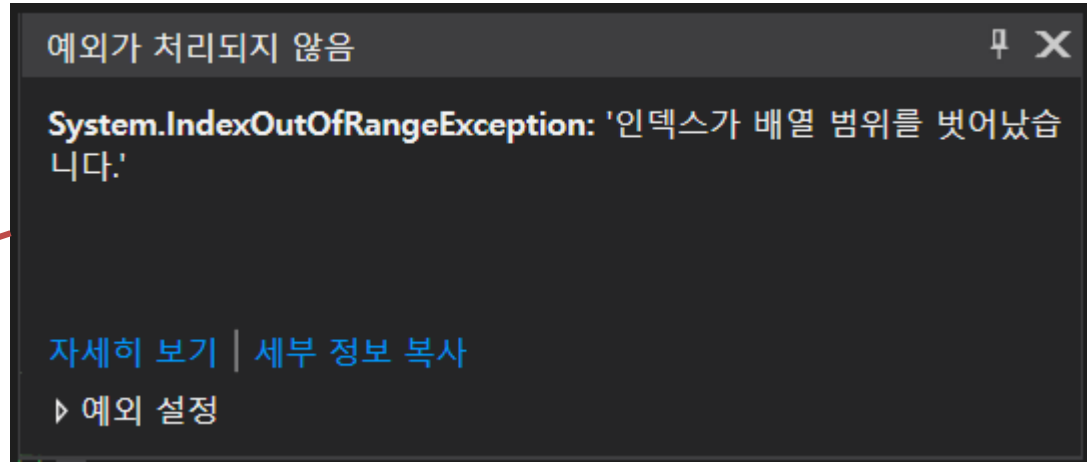
반복문과 배열

- length 속성
  - 배열의 요소 개수 확인

```
// 배열을 생성합니다.
int[] intArray = { 52, 48, 68, 157, 68, 32, 18 };

// 배열의 길이를 확인합니다.
int arrayLength = intArray.Length;
```

- IndexOutOfRangeException
  - 배열의 범위를 벗어나는 인덱스에 접근

```
// 배열을 생성합니다.
int[] intArray = { 52, 48, 68, 157, 68, 32, 18 };

// [7]배열에 있는 값을 출력합니다.
int arrayLength = intArray[7];
```



반복문과 배열

- 배열은 [0] 인덱스가 가장 처음
- 7개의 배열 생성 시 [0] ~ [6] 까지 존재

- 원하는 크기의 배열 생성 방법

```
int[] array = new int[100];
```

while 반복문

- 주어진 조건을 만족하면 명령문을 반복 수행한다.
- if 조건문과 형식은 비슷하나, 불 표현식이 참인 동안 중괄호 안의 문장 계속 실행
- 조건이 변하지 않으면 무한히 반복, 반드시 조건을 거짓으로 만드는 문장이 포함 되어야 함

```
while ([불 표현식])
{
   // 불 표현식이 참인 동안 실행할 문장
}
```

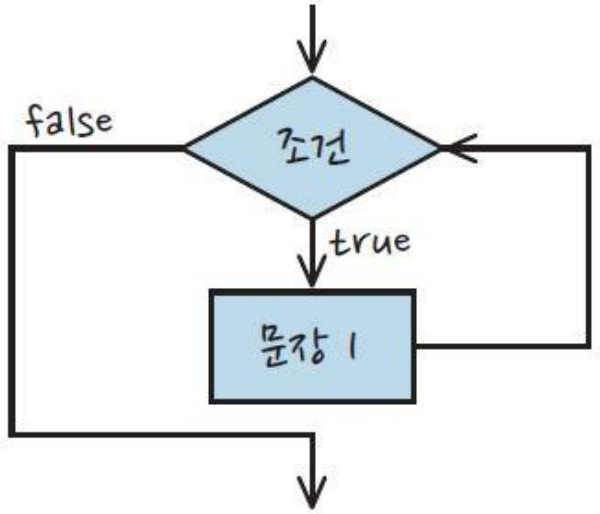


그림 4-4 while 반복문 형식과 순서도

while 반복문

- 실습 – while 문을 이용하여 임의의 수까지 합 구하기

1) 윈도우 폼 디자인

2) 속성 설정

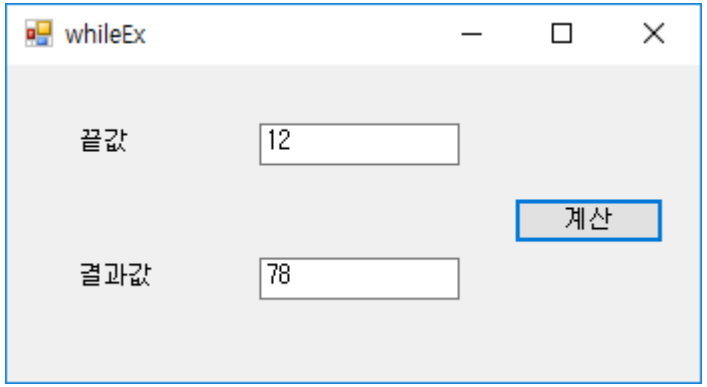
컨트롤	속성	속성값
Label1	Text	끝값
Label2	Text	결과값
TextBox1, TextBox2	TextAlign	Center
	Text	(빈칸)
Button1	Text	계산

while 반복문

3) 코드 작성

```
private void button1_Click(object sender, EventArgs e)
{
    int total, endValue, i;

    total = 0;
    i = 0;
    endValue = int.Parse(textBox1.Text);

    while (i < endValue)
    {
        total += ++i;

    }

    textBox2.Text = total.ToString();
}
```

4) 실행 결과



do while 반복문

- 문장을 먼저 실행 후 반복 조건을 검사하여 반복 여부 확인
- whlie 반복문과 형태는 비슷하나, 조건 비교 부분이 마지막에 위치

```
do
{
   // 불 표현식이 참인 동안 실행할 문장
} while ([불 표현식]);
```

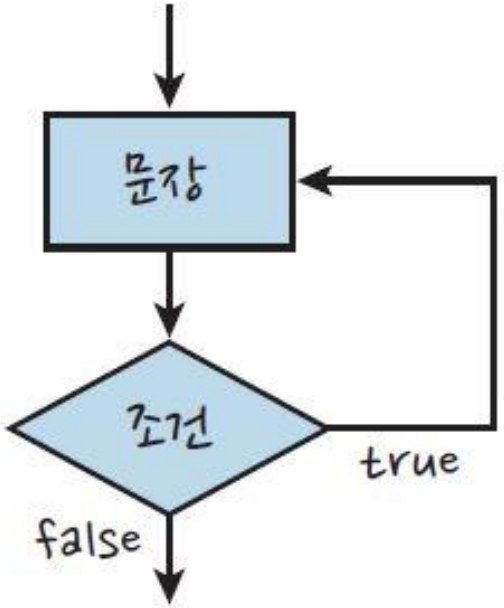


그림 4-5 do while 반복문 형식과 순서도

for 반복문

- 원하는 횟수만큼 반복 가능
- 맨 먼저 초기식 실행 후 조건식 확인
- for 반복문의 각 단계
  ① 초기식 비교
  ② 조건식 비교(조건 거짓이면 반복문 종료)
  ③ 문장을 실행
  ④ 종결식(증감식) 실행
  ⑤ 2단계로 이동

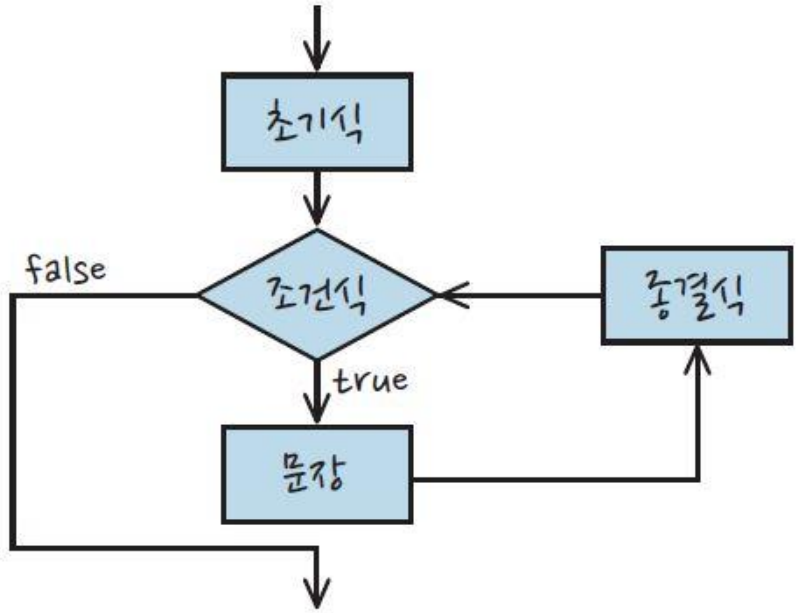


그림 4-6 for 반복문 순서도

```
for (int i = 0; i < [반복 횟수]; i++)
{

}
```

그림 4-7 for 반복문 형식

for 반복문

- 실습 – 정수 1부터 20 중에서의 5의 배수 합 구하기

1) 윈도우 폼 디자인

2) 속성 설정

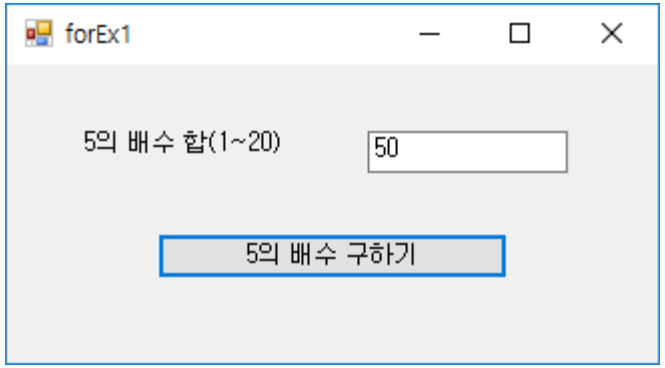
컨트롤	속성	속성값
Label1	Text	5의 배수 합(1~20)
Button1	Text	5의 배수 구하기
TextBox1	Text	(빈칸)

for 반복문

3) 코드 작성

```
private void button1_Click(object sender, EventArgs e)
{
    int sum = 0, i;

    for (i = 0; i <= 20; i += 5)
    {
        sum += i;
    }
    textBox1.Text = sum.ToString();

}
```

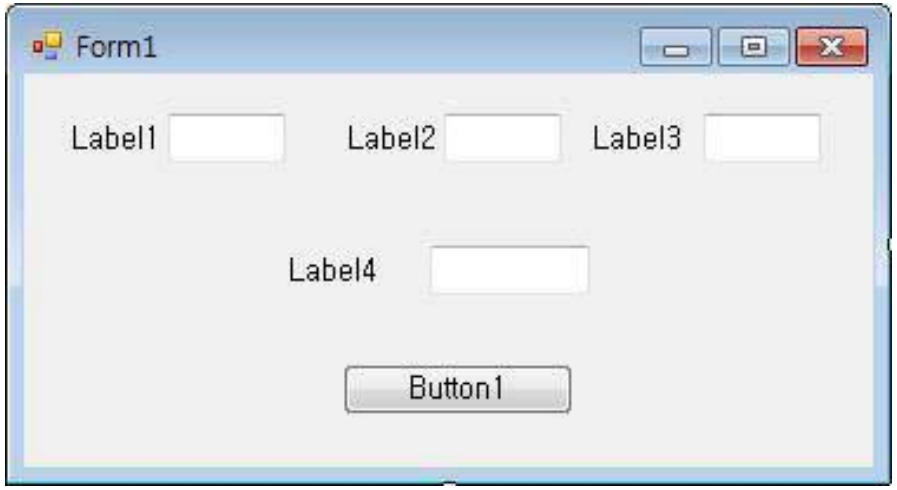
4) 실행 결과



for 반복문

- 실습 – 입력 받은 범위 내 숫자의 합 구하기

1) 윈도우 폼 디자인



2) 속성 설정

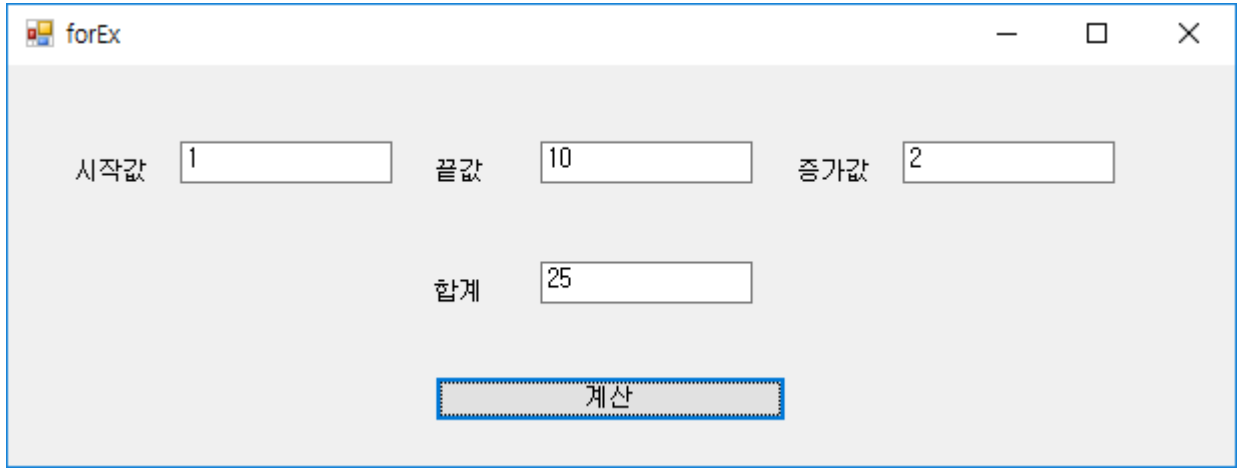
컨트롤	속성	속성값
Label1	Text	시작값
Label2	Text	끝값
Label3	Text	증가값
Label4	Text	합계
TextBox1~TextBox4	TextAlign	Center
	Text	(빈칸)
Button1	Text	계산

for 반복문

3) 코드 작성

```
private void button1_Click(object sender, EventArgs e)
{
    int start = int.Parse(textBox1.Text);
    int last = int.Parse(textBox2.Text);
    int add = int.Parse(textBox3.Text);
    int sum = 0;

    for(int i=start; i<last; i+=add)
    {
        sum += i;
    }
    textBox4.Text = sum.ToString();
}
```

4) 결과 확인



for 반복문

- 역 for 반복문
  - 배열 반복 뒤에서부터 실행 하는 경우 사용

```
for (int i = length - 1; i >= 0; i--)
{

}
```

forearch 반복문

- 반복문을 컬렉션에 쉽게 적용할 때 사용
  - 컬렉션 : 여러 개체가 모여서 집합을 이룬 것 (배열도 가능)

```
foreach (자료형 변수 in 컬렉션)
{

}
```

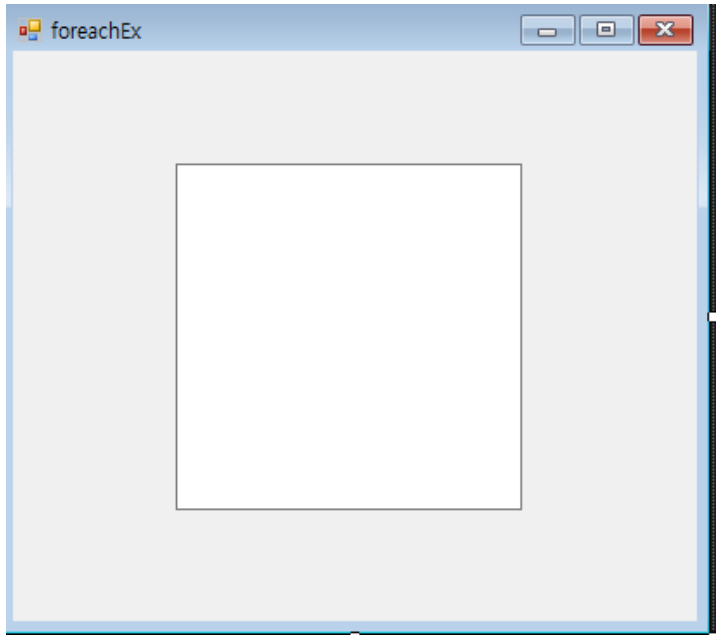
  - foreach 반복문 예

```
for (int i = 0; i < 컬렉션.길이; i++)
{
    자료형 변수 = 컬렉션[i];
}
```

forearch 반복문

- 실습 – 배열 값 출력하기

1) 윈도우 폼 디자인



2) 속성 설정

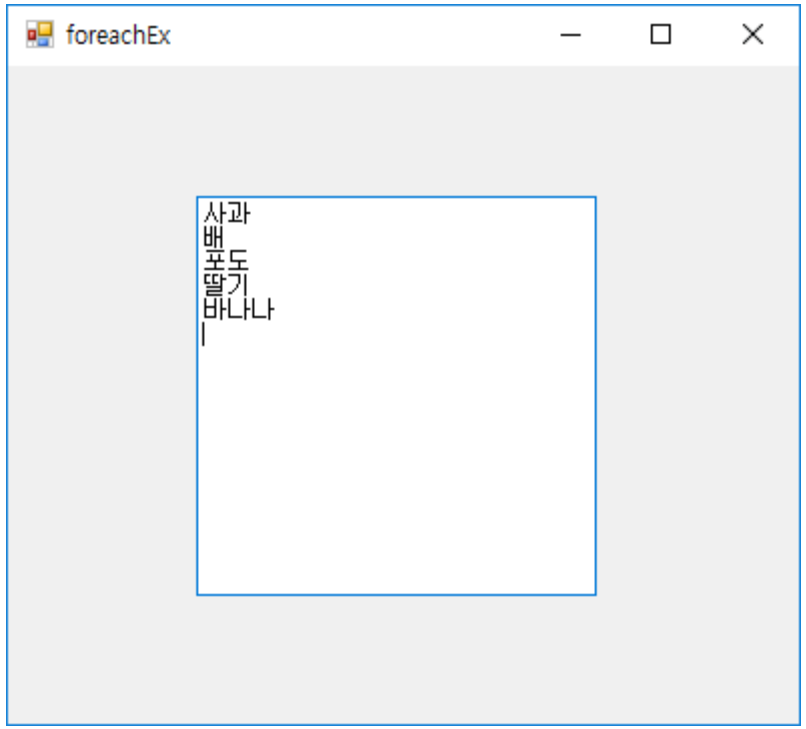
컨트롤	속성	속성값
textBox1	Multiline	True
	size	200, 200
	Text	(빈칸)

forearch 반복문

3) 코드 작성

```
private void foreachEx_Load(object sender, EventArgs e)
{
    string[] array = { "사과", "배", "포도", "딸기", "바나나" };

    foreach(var item in array)
    {
        textBox1.Text = textBox1.Text + item.ToString() +
            Environment.NewLine;
    }
}
```

4) 실행 결과



중첩 반복문

- 반복문을 여러 겹 중첩해서 사용하면 중첩 반복문
- 각 반복문의 초기식은 겹치면 안됨
  - ex)

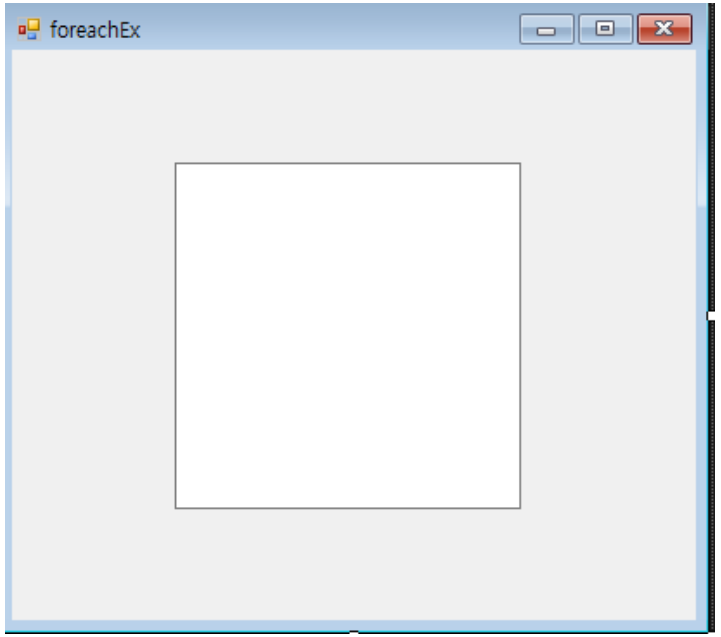
```
private void star_Load(object sender, EventArgs e)
{
    for(int i = 0; i < 10; i++)
    {
        for(int i=0; i< 10; i++)
        {

        }
    }
}
```

  - 위와 같이 같은 변수를 사용하면 오류가 발생하거나 실행이 제대로 되지 않을 수 있음

중첩 반복문

- 실습 – 별 피라미드 그리기

1) 윈도우 폼 디자인



2) 속성 설정

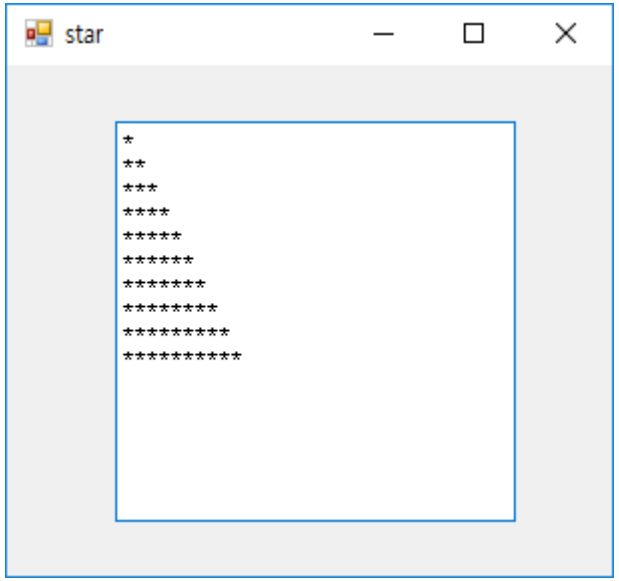
컨트롤	속성	속성값
textBox1	Multiline	True
	size	200, 200
	Text	(빈칸)

중첩 반복문

3) 코드 작성

```
private void star_Load(object sender, EventArgs e)
{
    for(int i = 0; i < 10; i++)
    {
        for(int j=0; j< i+1; j++)
        {
            textBox1.Text += "*";
        }
        textBox1.Text += Environment.NewLine;
    }
}
```

4) 실행 결과



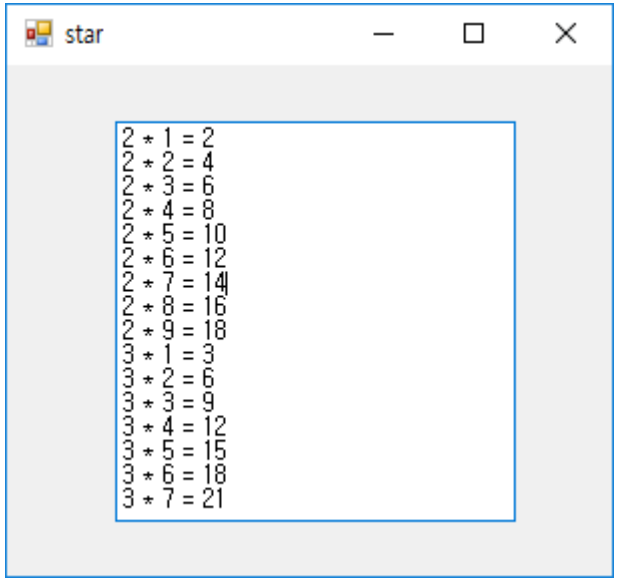
중첩 반복문

- 실습 – 구구단 작성

3) 코드 작성

```
private void star_Load(object sender, EventArgs e)
{
    // 구구단
    for(int i=2; i < 10; i++)
    {
        for(int j = 1; j < 10 ; j++)
        {
            textBox1.Text += i.ToString() + " * " + j.ToString() +
                " = " + (i*j).ToString();
            textBox1.Text += Environment.NewLine;
        }
    }
}
```

4) 실행 결과



break & continue 키워드

- break 키워드
  - switch 조건문 또는 반복문을 벗어날 때 사용하는 키워드
  - 무한 반복문을 빠져나갈 때 사용

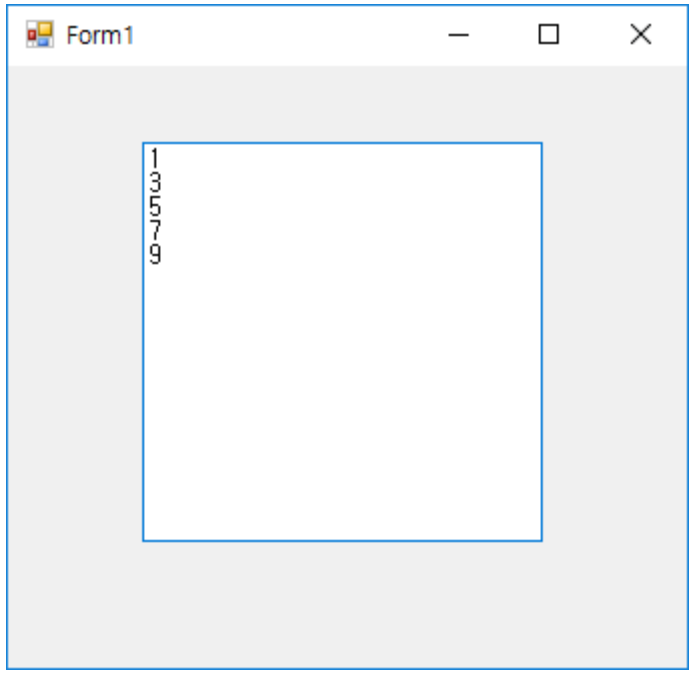
```
while (true)
{
    // 문장
}
```

  - 가장 가까운 중괄호를 빠져나감
  - 반복문 안에 반복문이 있을 시 하나만 빠져나갈 수 있음

break & continue 키워드

- continue 키워드
  - 반복문 내부에서 현재 반복을 멈추고 다음 반복을 진행
  - 예제

```
private void Form1_Load(object sender, EventArgs e)
{
    for(int i = 0; i < 10; i++)
    {
        if (i % 2 == 0)
            continue;
        textBox1.Text += i.ToString() + Environment.NewLine;
    }
}
```



  - 보통 if조건문과 같이 사용