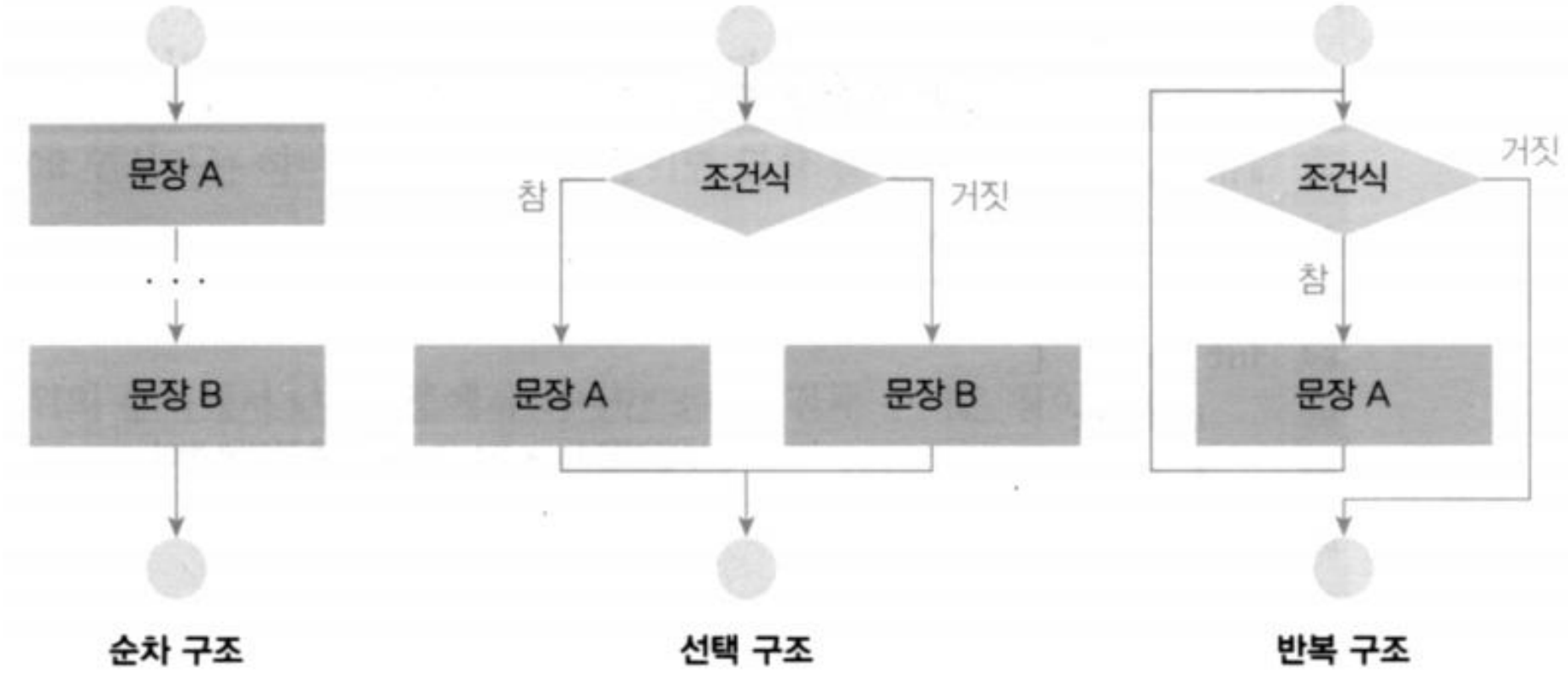
제어 구조와 배열

제어 구조

■ 제어 구조



제어 구조

■ 관계 연산자

연산자	의미
x == y	x와 y가 같은가?
x != y	x와 y가 다른가?
x 〉 y	x가 y보다 큰가?
x 〈 y	x가 y보다 작은가?
x 〉= y	x가 y보다 크거나 같은가?
x 〈= y	x가 y보다 작거나 같은가?

제어 구조

■ chapter02/ex01_rel.cpp] 관계 연산자

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

int main(int argc, char const *argv[]) {
  bool b;

  b = (1 == 2);

  cout<< std::boolalpha;  // 부울린을 true, false로 출력

  cout << b << endl;

  return 0;
}
```

```
false
```

제어 구조

■ chapter02/ex02_rel.cpp] 관계 연산자

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

int main(int argc, char const *argv[]) {
  bool b;
  int x = 3;
  int y = 3;
  cout<< std::boolalpha;  // 부울린을 true, false로 출력

  b = (x == 3) && (y==3);
  cout << b << endl;

  y = 2;
  b = (x == 3) && (y==3);
  cout << b << endl;
```

제어 구조

■ chapter02/ex02_rel.cpp] 관계 연산자

```
  b = (x == 3) || (y==3);
  cout << b << endl;

  x = 2;
  b = (x == 3) || (y==3);
  cout << b << endl;

  b = !(x==3);
  cout << b << endl;

  return 0;
}
```

```
true
false
true
false
true
```

제어 구조

■ 논리 연산자

연산자	의미
x && y	AND 연산, x와 y가 모두 참이면 참, 그렇지 않으면 거짓
x || y	OR 연산, x나 y중에서 하나만 참이면 참, 모두 거짓이면 거짓
!x	NOT 연산, x가 참이면 거짓, x가 거짓이면 참

제어 구조

■ if ~ else 문

```
if ( 조건식 ) {
    문장1
}
else {
    문장2
}
```

```
if( x == 100 )
    cout << "x가 100입니다." << endl;

if( x == 100 )
    cout << "x가 100입니다." << endl;
else
    cout << "x가 100이 아닙니다." << endl;
```

```
if( x == 100 ) {
    cout << "x의 값을 출력합니다." << endl;
    cout << "x가 100입니다." << endl;
}
```

제어 구조

■ chapter02/ex03_if.cpp] if 문

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

int main(int argc, char const *argv[]) {
  int x = 100;

  if(x == 100)
    cout << "x가 100입니다." << endl;

  if(x == 100) {
    cout << "x의 값을 출력합니다." << endl;
    cout << "x가 100입니다." << endl;
  }

  return 0;
}
```

```
x가 100입니다.
x의 값을 출력합니다.
x가 100입니다.
```

제어 구조

■ chapter02/ex04_if_else.cpp] if ~ else 문

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

int main(int argc, char const *argv[])
{
  int x = 120;

  if(x == 100)
    cout << "x가 100입니다." << endl;
  else
    cout << "x가 100이 아닙니다" << endl;
}
```

```
x가 100이 아닙니다
```

제어 구조

■ chapter02/ex05_if_else.cpp] if ~ else 문

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

int main(int argc, char const *argv[]) {
  int x, y;

  cout << "x값을 입력하세요 ";
  cin >> x;

  cout << "y값을 입력하세요 ";
  cin >> y;

  if(x > y)
    cout << "x가 y보다 큽니다." << endl;
  else
    cout << "y가 x보다 크거나 같습니다." << endl;
  return 0;
}
```

```
x값을 입력하세요30
y값을 입력하세요40
y가 x보다 크거나 같습니다.
```

제어 구조

■ 다중 if ~ else 문

```
if ( 조건식 ) {
    문장1
}
else if {
    문장2
}
else {
    문장3
}
```

```
if (x > 0)
    cout << "x는 양수입니다." << endl;
else if (x < 0)
    cout << "x는 음수입니다." << endl;
else
    cout << "x는 0입니다." << endl;
```

제어 구조

■ chapter02/ex06_multi_if.cpp] 다중 if ~ else 문

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

int main(int argc, char const *argv[]) {
  int x, y;

  cout << "x값을 입력하세요";
  cin >> x;

  cout << "y값을 입력하세요";
  cin >> y;

  if(x > y)
    cout << "x가 y보다 큽니다." << endl;
  else if (x < y)
    cout << "x가 y보다 작습니다." << endl;
  else
    cout << "x와 y가 같습니다." << endl;

  return 0;
}
```

```
x값을 입력하세요30
y값을 입력하세요40
y가 x보다 크거나 같습니다.
```

제어 구조

- switch 문

- chapter02/ex07_switch.cpp] switch 문

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

int main(int argc, char const *argv[]) {
  int number;

  cout << "숫자를 입력하세요:";
  cin >> number;
```

제어 구조

■ chapter02/ex07_switch.cpp] switch 문

```
  switch(number) {
    case 0:
      cout <<"zero\n";
      break;
    case 1:
      cout <<"one\n";
      break;
    case 2:
      cout <<"two\n";
      break;
    default:
      cout <<"many\n";
      break;
  }

  return 0;
}
```

```
숫자를 입력하세요:3
many
```

제어 구조

■ chapter02/ex08_switch.cpp] switch 문

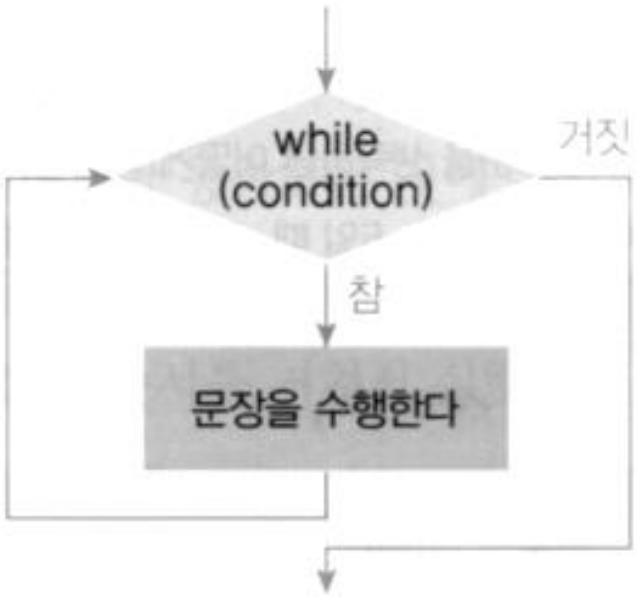
```
#include <iostream>
using namespace std;

int main(int argc, char const *argv[]) {
  int number;

  cout << "숫자를 입력하세요:";
  cin >> number;

  switch(number) {
    case 0:
      cout <<"zero\n";
    case 1:
      cout <<"one\n";
    case 2:
      cout <<"two\n";
    default:
      cout <<"many\n";
      break;
  }
  return 0;
}
```

```
숫자를 입력하세요:1
one
two
many
```

제어 구조

■ while 루프

○ 조건이 참일 동안 반복




```
while (조건식) {
    문장
}
```

제어 구조

■ chapter02/ex09_while.cpp] while 반복문

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

int main(int argc, char const *argv[]) {
  int n = 10;

  while(n>0) {
    cout << n << " ";
    n--;
  }

  cout << "fire!" << endl;
  return 0;
}
```

```
10 9 8 7 6 5 4 3 2 1 fire!
```

제어 구조

■ chapter02/ex10_while.cpp] while 반복문

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

int main(int argc, char const *argv[]) {
  int n;
  int i = 1;

  cout << "구구단 중에서 출력하고 싶은 단을 입력하세요: ";
  cin >> n;

  while( i<=9 ) {
    cout << n << " * " << i
         << " = " << n*i << endl;
    i++;
  }

  return 0;
}
```

제어 구조

■ while 반복문

```
구구단 중에서 출력하고 싶은 단을 입력하세요: 4
4 * 1 = 4
4 * 2 = 8
4 * 3 = 12
4 * 4 = 16
4 * 5 = 20
4 * 6 = 24
4 * 7 = 28
4 * 8 = 32
4 * 9 = 36
```

제어 구조

■ do ~ while 루프

○ 조건이 참일 동안 반복

```
do {
    문장
} while( 조건식 );
```

제어 구조

■ chapter02/ex11_do_while.cpp] do ~ while 반복문

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

int main(int argc, char const *argv[]) {
  string str;

  do {
    cout<<"문자열을 입력하세요:";
    getline(cin, str);

    cout << "사용자의 입력: " << str << endl;
  } while(str != "종료");

  return 0;
}
```

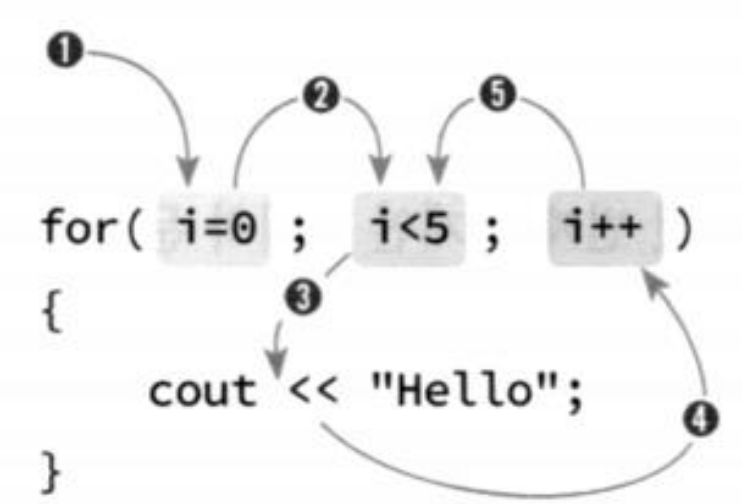
```
문자열을 입력하세요:안녕하세요
사용자의 입력: 안녕하세요
문자열을 입력하세요:종료
사용자의 입력: 종료
```

제어 구조

■ for 루프

○ 조건이 참일 동안 반복

```
for( 초기식 ; 조건식 ; 증감식 ) {
    문장
}
```



제어 구조

■ chapter02/ex12_for.cpp] for 반복문

```
#include <iostream>

using namespace std;

int main() {

  int sum = 0;
  for(int i=0; i<=10; i++) {
    sum += i;
  }

  cout << "1부터 10까지 정수의 합 = " << sum << endl;
  return 0;
}
```

```
1부터 10까지 정수의 합 = 55
```

제어 구조

■ chapter02/ex13_for.cpp] for 반복문

```
#include <iostream>

using namespace std;

int main() {
  long fact = 1;
  int n;

  cout << "정수를 입력하세요: ";
  cin >> n;

  for(int i=1; i<=n; i++) {
    fact = fact*i;
  }

  cout << n << "! = " << fact << endl;
  return 0;
}
```

```
정수를 입력하세요: 5
5! = 120
```

제어 구조

■ chapter02/ex14_break.cpp] break 문

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

int main() {
  for(int i=1; i<10; i++) {
    cout << i << " ";
    if(i==4) break;
  }

  return 0;
}
```

```
1 2 3 4
```

제어 구조

■ chapter02/ex15_continue.cpp] continue 문

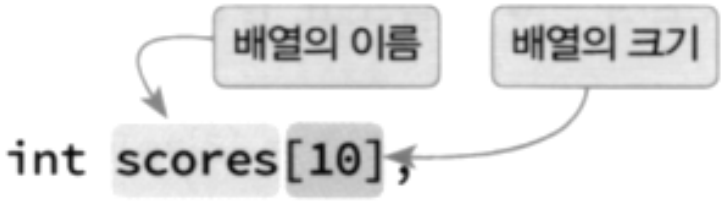
```
#include <iostream>

using namespace std;

int main() {
  for(int i=1; i<5; i++) {
    cout << "continue 문장 전에 있는 문장" << endl;
    continue;
    cout << "continue 문장 이후에 있는 문장" << endl;
  }

  return 0;
}
```

```
continue 문장 전에 있는 문장
continue 문장 전에 있는 문장
continue 문장 전에 있는 문장
continue 문장 전에 있는 문장
```

배열

■ 배열(array)

- ○ 같은 종류의 데이터들이 순차적으로 메모리에 저장되는 자료 구조
- ○ 각각의 데이터(요소)들은 인덱스(번호)를 사용하여 독립적으로 접근 가능
- ○ 대용량의 데이터를 동일한 이름으로 쉽게 저장하고 처리 가능




```
scores[5] = 80;
```

배열

■ chapter02/ex16_array.cpp] 배열

```
#include <iostream>

using namespace std;

int main() {

  const int STUDENTS = 10;

  int scores[STUDENTS];
  int sum = 0;
  int i, average;

  for(i=0; i<STUDENTS; i++) {
    cout << "학생들의 성적을 입력하시요: ";
    cin >> scores[i];
  }

  for(i=0; i<STUDENTS; i++) {
    sum += scores[i];
  }
```

배열

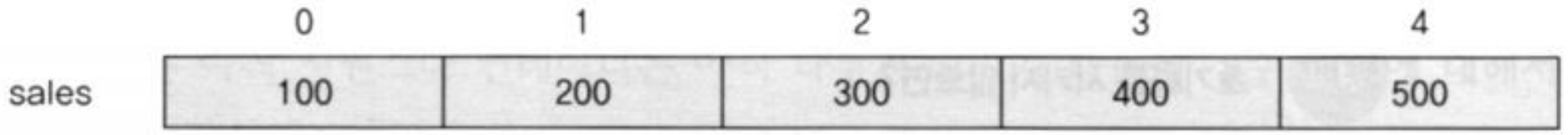
■ chapter02/ex16_array.cpp] 배열

```
  average = sum / STUDENTS;
  cout << "성적 평균= " << average << endl;

  return 0;
}
```

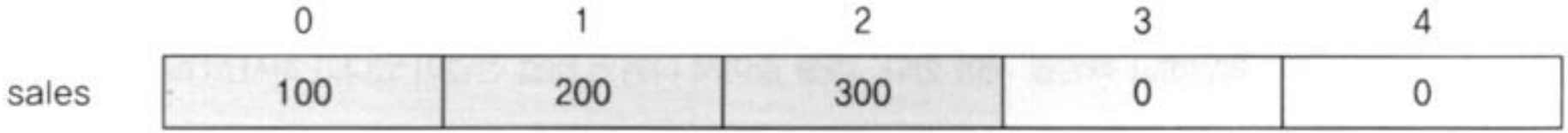
배열

■ 배열의 초기화

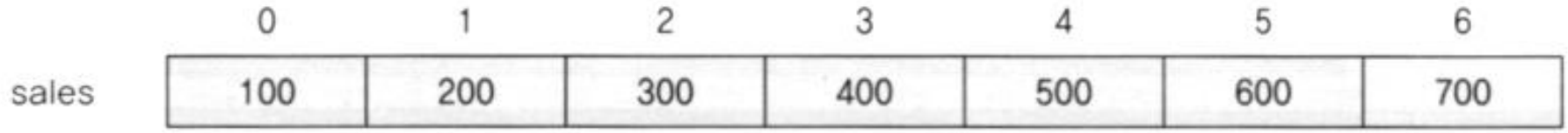
```
int sales[5] = { 100, 200, 300, 400, 500 };
```




```
int sales[5] = { 100, 200, 300 };
```




```
int sales[] = { 100, 200, 300, 400, 500, 600, 700 };
```



배열

■ 배열의 초기화

```
int scores[] = { 10, 20, 30 };
int scores[] { 10, 20, 30 };
```

```
int a{ 0 };          // int a=0;과 동일하다.
string s{ "hello" };         // string s="hello";
vector<string> list{ "alpha", "beta", "gamma" }; // 벡터 생성시 초기화
```

배열

■ chapter02/ex17_array_init.cpp] 배열의 초기화

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

int main() {
  const int STUDENTS = 5;

  int scores[STUDENTS] = {
    100, 200, 300, 400, 500
  };

  int sum = 0;
  int i, average;

  for(i=0; i<STUDENTS; i++) {
    sum += scores[i];
  }

  average = sum / STUDENTS;
  cout << "성적 평균= " << average << endl;

  return 0;
}
```

```
성적 평균= 300
```

배열

- 범위 기반 for 문

```
for( 변수 : 범위 ) {
    문장
}
```

제어 구조

■ chapter02/ex18_advanced_for.cpp] 범위 기반 for 문

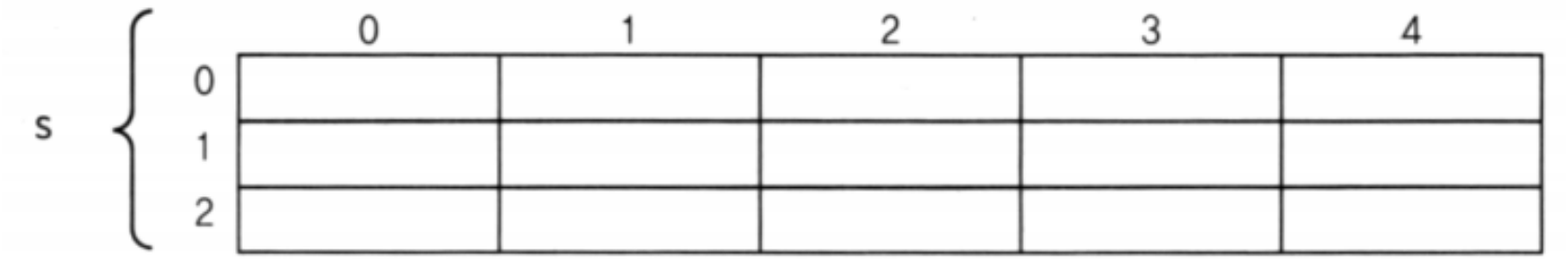
```
#include <iostream>
using namespace std;

int main() {
  int list[] = {1, 2, 3, 4, 5, 6 ,7, 8, 9, 10};
  int sum = 0;
  for(int i : list) {
    sum += i;
  }

  cout << sum << endl << endl;

  for(int& i : list) {
    cout << i << " ";
  }

  cout << endl;
  for(auto& i : list) {
    cout << i << " ";
  }
  return 0;
}
```

```
55

1 2 3 4 5 6 7 8 9 10
1 2 3 4 5 6 7 8 9 10
```

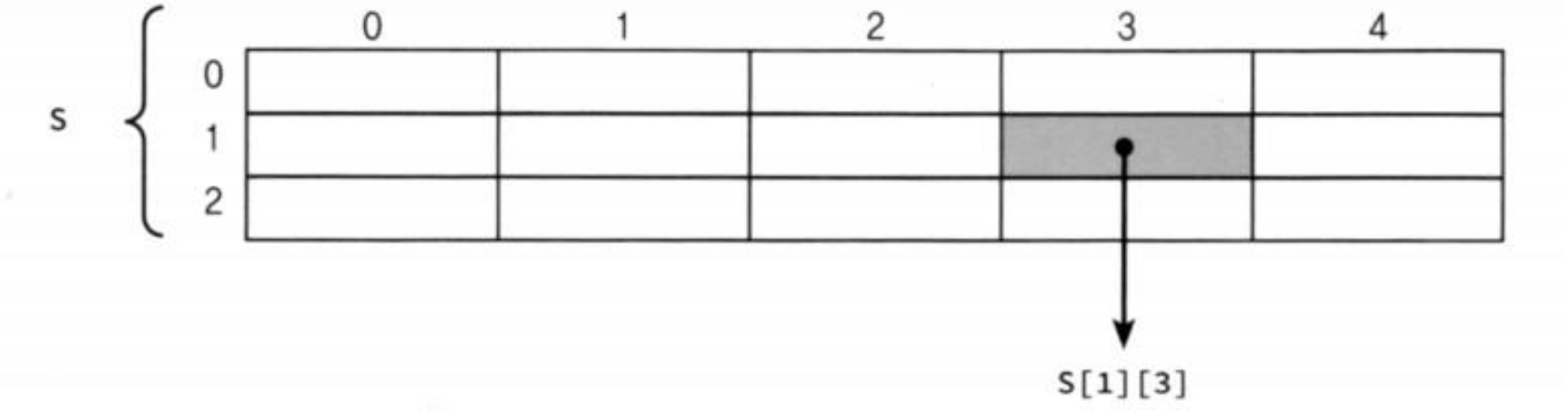
배열

■ 다 차원 배열

○ 2차원 배열




```
int s[3][5];
```



배열

■ 다 차원 배열의 초기화

```
int s[3][5] = {
    { 1, 2, 3, 4, 5 },        // 첫 번째 행의 요소들의 초기값
    { 2, 4, 6, 8, 10 },       // 두 번째 행의 요소들의 초기값
    { 3, 6, 9, 12, 15 }       // 세 번째 행의 요소들의 초기값
};
```

S

	0	1	2	3	4
0	1	2	3	4	5
1	2	4	6	8	10
2	3	6	9	12	15

배열

■ chapter02/ex19_muti_array.cpp] 다 차원 배열

```
#include <iostream>
using namespace std;
#define WIDTH 9
#define HEIGHT 3

int main() {
  int table[HEIGHT][WIDTH];
  int r, c;

  for(r = 0; r < HEIGHT; r++) {
    for(c=0; c < WIDTH; c++) {
      table[r][c] = (r+1)*(c+1);
    }
  }

  for(r = 0; r < HEIGHT; r++) {
    for(c=0; c < WIDTH; c++) {
      cout << table[r][c] << ", ";
    }
    cout << endl;
  }
  return 0;
}
```

```
1, 2, 3, 4, 5, 6, 7, 8, 9,
2, 4, 6, 8, 10, 12, 14, 16, 18,
3, 6, 9, 12, 15, 18, 21, 24, 27,
```