5장 배열

5.1 배열 소개

* 선언과 참조

- 배열(array): 동일한 자료형을 연속적으로 가진 변수리스트 이다.

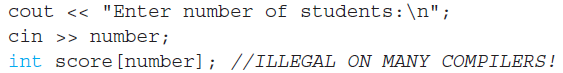
|  |
| --- |
| int score[5];  int score0, score1, score2, score3, score4; // 동일 효과 |

* + - 인덱스 변수, 첨자 변수, 배열 요소



* + - 5: 배열의 크기
    - 0, 1, …, 4: 인덱스, 첨자 – 첨자는 항상 0부터 시작하여 (배열 크기-1)까지 이다.

- 배열의 크기는 상수로 정의되어야 하며, 변수로 정의돼서는 안 된다. (컴파일러는 실행 전에 배열의 크기에 해당하는 공간을 잡아야 한다.)

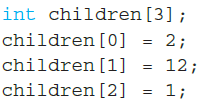


- 만약, 실행 중에 배열의 크기를 정하고자 한다면, 동적 배열을 사용해야 한다. (10장)

* 배열 초기화

- 배열 선언과 동시에 초기화 할 수 있다.





- 배열 인덱스의 수보다 적은 값으로 초기화되면, 처음부터 나열된 값으로 초기화되고 나머지는 0으로 할당된다.

- 배열을 선언하면서 초기화할 때 배열의 크기는 생략 가능하다.





* 메모리상의 배열

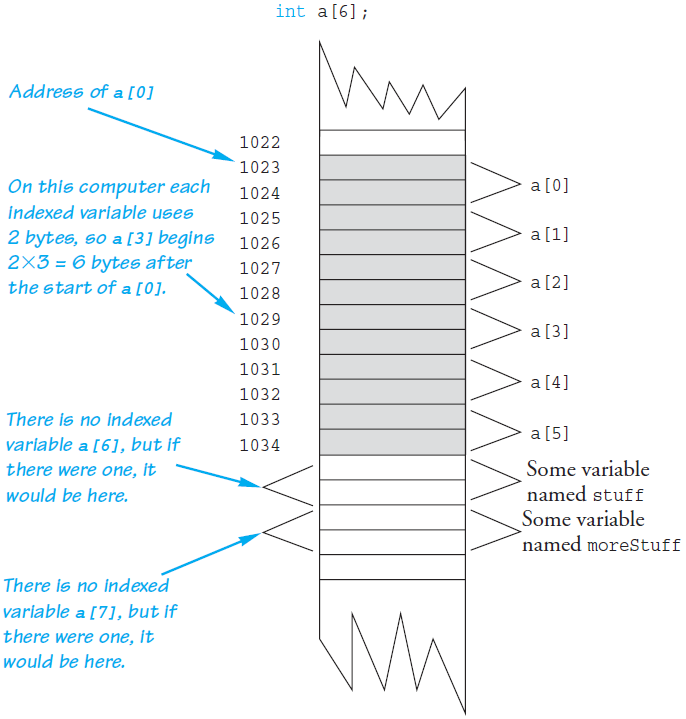
- 메모리는 데이터를 저장하는 부분과 이 저장 부분을 가리키는 주소 부분(address)으로 구성되어 있다.

- 배열은 3가지 즉, 첫 번째 배열 요소의 주소, 배열의 기본형, 배열의 크기가 중요하다.

- 데이터를 저장하는 부분은 자료형에 따라 그 크기가 바이트(byte) 단위로 달라진다.

- 배열명은 첫 번째 배열 요소의 주소를 가리킨다.

- (예제) ANSI C에서 int 형은 2 bytes 이상으로 정의된다. 따라서 int 형의 크기는 컴파일러가 설치된 각 플랫폼에(OS) 의해 결정된다.



예제: 최대값과 차이

ex05\_01.cpp

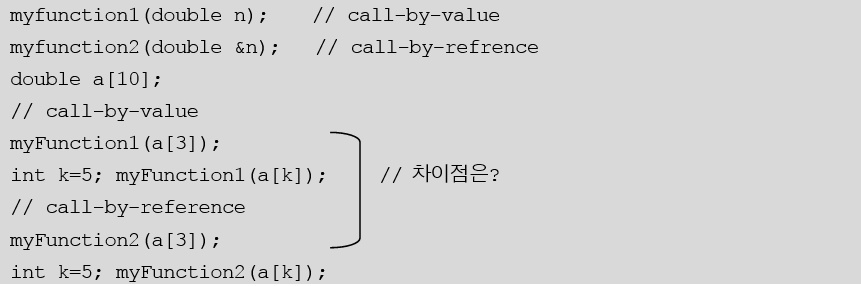
|  |
| --- |
| #include<iostream>  using namespace std;  int main()  {  int i, max, sc[5];  cout << "Enter 5 numbers: ";  for (i = 0; i < 5; i++)  cin >> sc[i];    max = sc[0];  for (i = 0; i < 5; i++)  if (sc[i] > max) max = sc[i];  cout << "MAX= " << max << "\n";  cout << "Defferences" << "\n";  for (i = 0; i < 5; i++)  cout << (max - sc[i]) << "\n";  return 0;  } |

5.2 함수에서의 배열

* 함수 인자로서의 배열 요소(인덱스 변수)

- 배열의 각 원소가 인자로 함수에 전달된다.

- 인덱스 변수는 call-by-value나 call-by-reference 인자가 모두 가능하다.



예제: 배열의 각 원소를 함수에 전달

ex05\_02.cpp

|  |
| --- |
| #include<iostream>  using namespace std;  void print\_defference(int max, int data);  int main()  {  int i, max, sc[5];  cout << "Enter 5 numbers: ";  for (i = 0; i < 5; i++)  cin >> sc[i];    max = sc[0];  for (i = 0; i < 5; i++)  if (sc[i] > max) max = sc[i];  cout << "MAX= " << max << "\n";  cout << "Defferences" << "\n";  //인자로서의 각 배열  for (i = 0; i < 5; i++)  print\_defference(max,sc[i]);  return 0;  }  void print\_defference(int max, int data)  {  cout << (max - data) << "\n";  } |

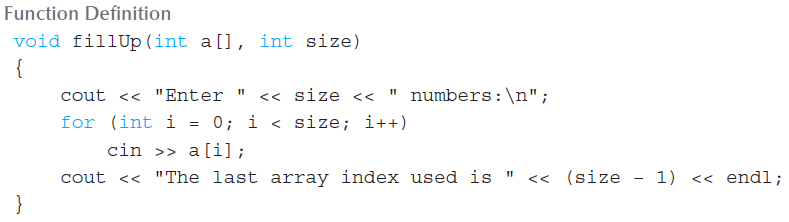
* 함수 인자로서의 전체 배열

- 배열 매개변수(array parameter): 함수가 호출될 때, 배열 전체가 형식 매개변수에 전달된다. - C++의 새로운 특징

- call-by-value나 call-by-reference도 아닌 새로운 형이지만, call-by-reference와 유사하게 동작한다.

- []가 배열 매개변수를 의미한다.





- 함수의 정의나 헤더에 a[정수]와 같이 숫자를 넣을 수 있지만, 컴파일러는 숫자를 무시한다. 따라서, a[정수]=a[]는 동일하다.

- 함수 호출 시에는 어떠한 대괄호([])나 식을 포함하지 않는다.

주의 사항

- 배열이 함수의 인자로 사용될 때, 배열의 첫 번째 요소의 주소만 함수에게 알려준다. 따라서 call-by-pointer도 이용할 수 있다.

- 배열의 기본형은 형식 매개 변수의 형과 일치해야 하므로, 함수는 배열의 기본 형을 알고 있다.

- 그러나 배열의 크기는 함수에게 알려주지 않는다. ex) fun(int a[])

- 배열의 기본형과 크기가 분리되어 있기 때문에 같은 함수로 다른 크기의 배열 인자에 대해서 사용할 수 있다.

예제: 전체 배열을 함수에 전달

ex05\_03.cpp

|  |
| --- |
| #include<iostream>  using namespace std;  int find\_max(int ary[], int size);  //int find\_max(int\* ary, int size);  void print\_defference(int max, int data);  int main()  {  int i, max, sc[5];  cout << "Enter 5 numbers: ";  for (i = 0; i < 5; i++)  cin >> sc[i];    //인자로서 전체 배열  max = find\_max(sc, 5);  cout << "MAX= " << max << "\n";  cout << "Defferences" << "\n";  //인자로서의 각 배열 요소  for (i = 0; i < 5; i++)  print\_defference(max,sc[i]);  return 0;  }  //int find\_max(int \*ary, int size)  int find\_max(int ary[10], int size)  {  int max;  max = ary[0];  for (int i = 0; i < 5; i++)  if (ary[i] > max) max = ary[i];  return max;  }  void print\_defference(int max, int data)  {  cout << (max - data) << "\n";  } |

포인터를 이용한 전체 배열 전달

- 배열의 이름이 첫 번째 요소의 주소를 가리키고 있으므로, 함수의 매개변수를 포인터를 이용하여 구현할 수 있다.

예제: 포인터를 이용하여 전체 배열을 함수에 전달

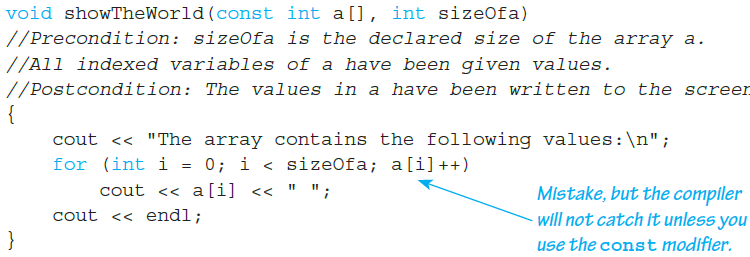
ex05\_03.cpp에서 다음을 수정

|  |
| --- |
| int find\_max(int \*array, int size); |

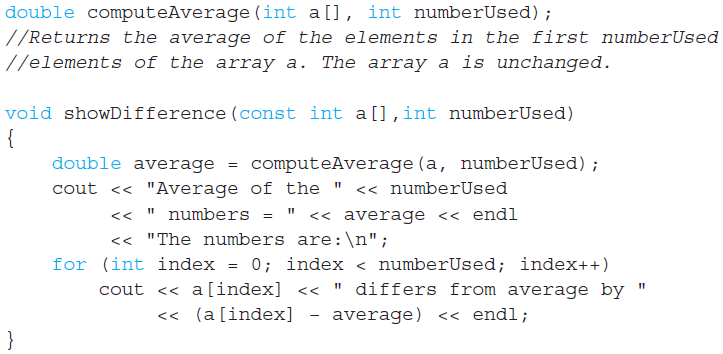
* const 매개변수 수정자

- 함수 호출에서 배열 매개변수(a[])를 사용할 때 함수는 배열에 저장된 값을 변경할 수 있다.

- 컴파일러에게 배열 인자의 변경을 원하지 않는다고 알리기 위해 배열 매개변수 앞에 const 수정자를 삽입한다.



- const 수정자는 일관성 있게 사용되어야 한다. 특정 형의 배열 매개변수에 대해 const를 사용한다면, 같은 형을 가진 배열 매개변수가 다른 함수에 의해 변경되지 않도록 const를 사용해야 한다. 즉 상수는 항상 상수이어야 한다.



- 위 코드는 경고 메시지를 출력한다. 비록 computeAverage 함수는 a[]의 값을 변경시키지 않지만, 컴파일러는 a[]가 변경될 가능성이 있기 때문에 경고를 출력한다.

5.3 배열 프로그래밍

- 부분적으로 채워진 배열은 사용할 때 주의를 기울여야 한다. 프로그램은 배열의 얼마나 많은 부분이 사용되는지 알고 있어야 하며 값이 할당되지 않은 배열 인덱스에 대해서는 접근하지 말아야 한다.

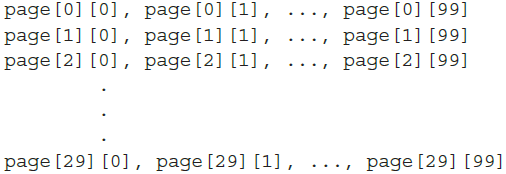
- 선택 정렬: 자료 구조 시간에 학습

5.4 다차원 배열

* 다차원 배열의 기본

- 2차원 배열은 첫 번째 인덱스는 **행**, 두 번째 인덱스는 **열**로 나타낼 수 있다.





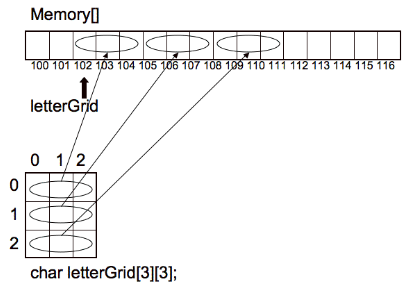
* 다차원 배열의 함수 매개변수

- 첫 번째 차원의 크기는 1차원 배열처럼 주어지지 않아도 된다(함수 매개 변수에서). 이것은 배열의 첫 번째 요소가 주소를 나타내기 때문이다.

- 두 번째 차원부터는 그 크기가 반드시 주어져야 한다.

- 첫 번째 차원의 크기를 알려주기 위해, 함수에게 정수형 매개변수를 추가한다.

|  |
| --- |
| void getPage(char letterGrid[][3], int firstsize); // 3byte 크기의  // char형이 firstzise 만큼 있다. |



예제: 다차원 배열의 함수 전달

ex05\_04.cpp

|  |
| --- |
| #include<iostream>  using namespace std;  //void find\_max(int array[][4], int fmax[], int row);  void find\_max(int array[][4], int fmax[], int row, int column);  void print\_difference(int max, int data);  int main()  {  int sc[2][4] = { {5,6,7,8},{13,12,11,15} };  int max[2];    //인자로서 전체 배열  //find\_max(sc, max, 2);  find\_max(sc, max, 2, 4);  cout << "Max0 = " << max[0] << endl;  cout << "Max1 = " << max[1] << endl;    cout << "Differences" << "\n";  for(int i = 0; i<2; i++)  for (int j = 0; j < 4; j++)  {  print\_difference(max[i], sc[i][j]);  }  //주소값 확인  cout << sc << endl;  cout << &(sc[0][0]) << endl;  return 0;  }  //void find\_max(int array[][4], int fmax[], int row  void find\_max(int array[][4], int fmax[], int row, int column)  {  for (int i = 0; i < row; i++)  {  fmax[i] = array[i][0];  //for (int j = 0; j < 4; j++)  for(int j = 0; j <column; j++)  if (array[i][j] > fmax[i]) fmax[i] = array[i][j];  }  }  void print\_difference(int max, int data)  {  cout << (max - data) << "\n";  } |