|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 2018\_2\_C++ \_12 | 학번 : |  | 이름 : |  |

1. 제네릭 함수

|  |
| --- |
| #include <iostream>  using namespace std;  template <class T>  void print(T array [], int n) {  for(int i=0; i<n; i++)  cout << array[i] << '\t';  cout << endl;  }  void print(char array [], int n) { // 템플릿 함수와 동일한 이름의 함수 중복  for(int i=0; i<n; i++)  cout << (int)array[i] << '\t'; // array[i]를 int 타입으로 변환하여 정수 출력  cout << endl;  }  int main() {  int x[] = {1,2,3,4,5};  double d[5] = { 1.1, 2.2, 3.3, 4.4, 5.5 };  print(x, 5); // 템플릿으로부터 구체화한 함수 호출  print(d, 5); // 템플릿으로부터 구체화한 함수 호출  char c[5] = {1,2,3,4,5};  print(c, 5); // char 배열을 숫자로 출력하는 중복 함수 호출  } |
| **[실행 결과]** |

* 제네릭 클래스

|  |
| --- |
| #include <iostream>  using namespace std;  template <class T1, class T2> // 두 개의 제네릭 타입 선언  class GClass {  T1 data1;  T2 data2;  public:  GClass();  void set(T1 a, T2 b);  void get(T1 &a, T2 &b);  };  template <class T1, class T2>  GClass<T1, T2>::GClass() {  data1 = 0; data2 = 0;  }  template <class T1, class T2>  void GClass<T1, T2>::set(T1 a, T2 b) {  data1 = a; data2 = b;  }  template <class T1, class T2>  void GClass<T1, T2>::get(T1 & a, T2 & b) {  a = data1; b = data2;  }  int main() {  int a;  double b;  GClass<int, double> x;  x.set(2, 0.5); // int, double의 매개 변수  x.get(a, b);  cout << "a=" << a << '\t' << "b=" << b << endl;  char c;  float d;  GClass<char, float> y;  y.set('m', 12.5); // char, double의 매개 변수  y.get(c, d);  cout << "c=" << c << '\t' << "d=" << d << endl;  } |
| **[실행 결과]** |

* STL(Standard Template Library 활용 - vector 컨테이너

|  |
| --- |
| #include <iostream>  #include <string>  #include <vector>  using namespace std;  int main() {  vector<string> sv; // 문자열 벡터 생성  string name;  cout << "이름을 5개 입력하라" << endl;  for (int i = 0; i<5; i++) { // 한 줄에 한 개씩 5 개의 이름을 입력받는다.  cout << i + 1 << ">>";  getline(cin, name);  sv.push\_back(name);  }  name = sv.at(0); // 벡터의 첫 번째 이름  for (int i = 1; i<sv.size(); i++) {  if (name < sv[i]) // sv[i]의 문자열이 name보다 사전에서 뒤에 나옴  name = sv[i]; // name을 sv[i]의 문자열로 변경  }  cout << "사전에서 가장 뒤에 나오는 이름은 " << name << endl;  } |
| **[실행 결과]** |

* STL(Standard Template Library 활용 - iterator 사용

|  |
| --- |
| #include <iostream>  #include <vector>  using namespace std;  int main() {  vector<int> v; //정수 벡터 생성  v.push\_back(1);  v.push\_back(2);  v.push\_back(3);  vector<int>::iterator it; // 벡터 v의 원소에 대한 포인터 it 선언  for (it = v.begin(); it != v.end(); it++) { // iterator를 이용하여 모든 원소 탐색  int n = \*it; // it가 가리키는 원소 값 리턴  n = n \* 2; // 곱하기 2  \*it = n; // it가 가리키는 원소에 값 쓰기  }  for (it = v.begin(); it != v.end(); it++) // 벡터 v의 모든 원소 출력  cout << \*it << ' ';  cout << endl;  } |
| **[실행 결과]** |

* 내용 점검

1. 교재 10장 연습문제

|  |  |
| --- | --- |
| 번호 | 문제풀이 |
| 1 |  |
| 2 |  |
| 3 |  |
| 4 |  |
| 5 |  |
| 6 |  |
| 7 |  |
| 8 |  |
| 9 |  |
| 10 |  |
| 11 |  |
| 12 |  |

* 프로그램 과제

1. 교재 525p : 문제 2번

|  |
| --- |
| **[프로그램 소스]**  #include <iostream>  using namespace std;  template <class T>  bool equalArrays(T\* p, T\* q, int n) {  for(int i=0; i<n; i++)  if(p[i] != q[i]) return false;  return true;  }  int main() {  int x[] = {1, 10, 100, 5, 4};  int y[] = {1, 10, 100, 5, 4};  double a[] = { 0.1, 2.3, 3.5};  double b[] = { 5.3, -2.1, 12.5};  if(equalArrays(x, y, 5)) cout << "같다" << endl; // 5는 배열 x, y의 크기  else cout << "다르다" << endl;  if(equalArrays(a, b, 3)) cout << "같다" << endl; // 3은 배열 a, b의 크기  else cout << "다르다" << endl;  } |
| **[실행 결과]** |

1. 교재 525p : 문제 3번

|  |
| --- |
| **[프로그램 소스]**  include <iostream>  using namespace std;  template <class T>  void reverseArray(T\* p, int n) {  for(int i=0; i<n/2; i++) {  T tmp = p[i];  p[i] = p[n-i-1];  p[n-i-1] = tmp;  }  return;  }  int main() {  int x[] = {1, 10, 100, 5, 4};  reverseArray(x, 5);  for(int i=0; i<5; i++)  cout << x[i] << ' ';  cout << endl;  int y[] = {-1, -10, -100, -1000};  reverseArray(y, 4);  for(int i=0; i<4; i++)  cout << y[i] << ' ';  cout << endl;  } |
| **[실행 결과]** |

1. 교재 525p : 문제 4번

|  |
| --- |
| **[프로그램 소스]**  #include <iostream>  using namespace std;  template<class T>  bool search(T element, T ar[], int n) {  for(int i=0; i<n; i++) {  if(ar[i] == element)  return true;  }  return false;  }  int main() {  int x[] = {1, 10, 100, 5, 4};  if(search(100, x, 5)) // 100이 배열 x에 포함되어 있는가?  cout << "100이 배열 x에 포함되어 있다";  else  cout << "100이 배열 x에 포함되어 있지 않다";  cout << endl;  char c[] = {'h', 'e', 'l', 'l', 'o'};  if(search('e', c, 5)) // 'e'가 배열 c에 포함되어 있는가?  cout << "e가 배열 c에 포함되어 있다";  else  cout << "e가 배열 c에 포함되어 있지 않다";  cout << endl;  } |
| **[실행 결과]** |

1. 교재 527p : 문제 8번

|  |
| --- |
| **[프로그램 소스]**  #include <iostream>  using namespace std;  class Comparable {  public:  virtual bool operator > (Comparable& op2)=0;  virtual bool operator < (Comparable& op2)=0;  virtual bool operator == (Comparable& op2)=0;  };  class Circle : public Comparable {  int radius;  public:  Circle(int radius=1) { this->radius = radius; }  int getRadius() { return radius;}  virtual bool operator > (Comparable& op2) {  Circle \*p = (Circle\*)&op2;  if(radius > p->radius) return true;  return false;  }  virtual bool operator < (Comparable& op2) {  Circle \*p = (Circle\*)&op2;  if(radius < p->radius) return true;  return false;  }  virtual bool operator == (Comparable& op2) {  Circle \*p = (Circle\*)&op2;  if(radius == p->radius) return true;  return false;  }  };  template <class T>  T bigger(T a, T b) { // 두 개의 매개 변수를 비교하여 큰 값을 리턴  if(a > b) return a;  else return b;  }  int main() {  int a=20, b=50, c;  c = bigger(a, b);  cout << "20과 50중 큰 값은 " << c << endl;  Circle waffle(10), pizza(20), y;  y = bigger(waffle, pizza);  cout << "waffle과 pizza 중 큰 것의 반지름은 " << y.getRadius() << endl;  } |
| **[실행 결과]** |

1. 교재 527p : 문제 9번

|  |
| --- |
| **[프로그램 소스]**  #include <iostream>  #include <vector>  using namespace std;  void printVector(const vector<int> &v) {  int n = v.size();  // 벡터의 각 원소 출력  for(int i=0; i<n; i++)  cout << v[i] << ' ';  cout << endl;  }  double getAverage(const vector<int> &v) {  int sum = 0;  int n = v.size();  // 벡터의 원소 개수 모두 더하기  for(int i=0; i<n; i++)  sum+=v[i];  return double(sum)/n;  }  int main() {  vector<int> v; // 벡터 객체 생성  while(true) {  cout << "정수를 입력하세요(0을 입력하면 종료)>>";  int num;  cin >> num;  if(num == 0)  return 0; // 프로그램 종료    v.push\_back(num); //입력받은 수를 벡터에 저장  printVector(v); // 벡터의 모든 수 출력  cout << "평균 = " << getAverage(v) << endl; // 평균 계산 후 출력  }  } |
| **[실행 결과]** |