**202211341 컴퓨터공학부 이윤희**

|  |
| --- |
| **Question1** |
| **Source code** |
| #include<iostream>  #include<string>  using namespace std;  template <typename T, int N>  void reverse(T(&array)[N]) {  for (int i = 0; i < (N/2); i++) {  swap(array[i], array[N - i - 1]);  }  }  template <typename T, int N>  void print(T(&array)[N]) {  for (int i = 0; i < N; i++) {  cout << array[i] << " ";  }  cout << endl;  }  int main(void) {  int iarr[5] = { 3,7,2,12,14 };  double darr[5] = { 22.7,14.2,3.8,12.23,11.2 };  char carr[6] = { 'C','a','B','E','N','Q' };  string sarr[4] = { "John","Lu","Mary","Su" };  bool barr[8] = { true,false,true,true,false,false,false,true };  long long larr[3] = { 9223372036854775807,223372036854775807 , 7654372036854775807 };  cout << "Original array" << endl;  print(iarr); reverse(iarr);  cout << "Reversed array" << endl;  print(iarr); cout << endl;  cout << "Original array" << endl;  print(darr); reverse(darr);  cout << "Reversed array" << endl;  print(darr); cout << endl;  cout << "Original array" << endl;  print(carr); reverse(carr);  cout << "Reversed array" << endl;  print(carr); cout << endl;  cout << "Original array" << endl;  print(sarr); reverse(sarr);  cout << "Reversed array" << endl;  print(sarr); cout << endl;  cout << "Original array" << endl;  print(barr); reverse(barr);  cout << "Reversed array" << endl;  print(barr); cout << endl;  cout << "Original array" << endl;  print(larr); reverse(larr);  cout << "Reversed array" << endl;  print(larr); cout << endl;  return 0;  } |
| **Output** |
| **Question2** |
| **Source code** |
| #include<iostream>  #include<string>  #include<stdexcept>  using namespace std;  class ArrIndexException :public out\_of\_range {  public:  ArrIndexException(const string&);  };  ArrIndexException::ArrIndexException(const string& m)  :out\_of\_range(m){  }  ArrIndexException indexex("Array is full.\n");  template<typename T>  class Array {  public:  T\* arr;  int N;  int size;  public:  Array(int);  ~Array();  void add(T);  void print();  };  template<typename T>  Array<T>::Array(int n):N(n) {  arr = new T[n];  size = 0;  }  template<typename T>  Array<T>::~Array() {  }  template<typename T>  void Array<T>::add(T data) {  try {  if (size<N) {  arr[size++] = data;  }  else {  throw indexex;  }  }  catch (...) {  throw;  }  }  template<typename T>  void Array<T>::print() {  for (int i = 0; i < size; i++) {  cout << arr[i] << " ";  }  cout << endl;  }  int main(void) {  cout << "Instantiation of an array of integers." << endl;  Array<int> array1(5);  try {  array1.add(-5);  array1.add(7);  array1.add(8);  array1.add(10);  array1.add(14);  array1.add(20);  array1.add(-14);  }  catch (out\_of\_range& ex) {  cout << ex.what();  }  array1.print(); cout << endl;    cout << "Instantiation of an array of doubles." << endl;  Array<double> array2(5);  try {  array2.add(5.3);  array2.add(7.6);  array2.add(8.1);  array2.add(-1);  }  catch (out\_of\_range& ex) {  cout << ex.what();  }  array2.print(); cout << endl;  cout << "Instantiation of an array of strings." << endl;  Array<string> array3(4);  try {  array3.add("John");  array3.add("Lu");  array3.add("Mary");  array3.add("Leo");  array3.add("Robert");  }  catch (out\_of\_range& ex) {  cout << ex.what();  }  array3.print(); cout << endl;  cout << "Instantiation of an array of characters." << endl;  Array<char> array4(11);  try {  array4.add('p'); array4.add('r'); array4.add('o');  array4.add('g'); array4.add('r'); array4.add('a');  array4.add('m'); array4.add('m'); array4.add('i');  array4.add('n'); array4.add('g'); array4.add('s');  }  catch (out\_of\_range& ex) {  cout << ex.what();  }  array4.print(); cout << endl;    cout << "Instantiation of an array of booleans." << endl;  Array<bool> array5(30);  try {  array5.add(true); array5.add(true); array5.add(false);  array5.add(true); array5.add(true); array5.add(false);  array5.add(false); array5.add(false); array5.add(false);  array5.add(true); array5.add(false); array5.add(false);  array5.add(false); array5.add(true); array5.add(false); array5.add(true); array5.add(true); array5.add(false);  array5.add(true); array5.add(true); array5.add(false);  array5.add(false); array5.add(false); array5.add(false);  array5.add(true); array5.add(false); array5.add(false);  array5.add(true); array5.add(true); array5.add(false);  array5.add(true); array5.add(true); array5.add(false);  array5.add(false); array5.add(false); array5.add(false);  }  catch (out\_of\_range& ex) {  cout << ex.what();  }  array5.print(); cout << endl;  return 0;  } |
| **Output** |