**CPP Problem Design Example**

|  |
| --- |
| **Subject: Template** |
| **Contributor: 鍾賢廣** |
| **Main testing concept: structure/class**   |  |  | | --- | --- | | **Basics** | **Functions** | | ■ C++ BASICS  ■ FLOW OF CONTROL  ■ FUNCTION BASICS  □ PARAMETERS AND OVERLOADING  □ ARRAYS  □ STRUCTURES AND CLASSES  □ CONSTRUCTORS AND OTHER TOOLS  □ OPERATOR OVERLOADING, FRIENDS, AND REFERENCES  □ STRINGS  □ POINTERS AND DYNAMIC ARRAYS | □ SEPARATE COMPILATION AND NAMESPACES  □ STREAMS AND FILE I/O  □ RECURSION  □ INHERITANCE  □ POLYMORPHISM AND VIRTUAL FUNCTIONS  □ TEMPLATES  □ LINKED DATA STRUCTURES  □ EXCEPTION HANDLING  □ STANDARD TEMPLATE LIBRARY  □ PATTERNS AND UML | |
| **Description:**  Create a template Array<T>, which has member functions and member variables:  Array(int newLength): create an array and is’s length is newLength. In addition, allocate a block of memory for an array of *newLength* elements, each of them initializes all its bits to zero.  void get(int index): Return the element at position *index* in the array. If there is an exception, you need to throw the message “the index is out of array”.  *void set(int index, T value)*: Set the element at position *index* in the array. If there is an exception, you need to throw the message “the index is out of array”.  *void clear()*: Clear the array. Note: *length* will be zero.  *ReadOnly<int>* ***length***: Length of this *Array*.  Also,  Create a class Exception as exception object with a member function message () which outputs the exception message.  Create a class template ReadOnly<T> which constrains objects read-only (e.g. data member length of Array is read-only) and has member functions:  to initialize its data member.  to throw the message “you can not change this value” if you try to write or assign a value to length after construction.  to printout its data member.  Here is an use case of ReadOnly<T>:  ReadOnly<int> integer(3);  std::cout << integer; // will print 3  try {  integer = 5; // error. The message is “you can not change this value”.  }  catch (Exception e) {  std::cout << e.message() << std::endl;  }  **Input:**  There are no further inputs, please refers to the example.cpp files.  **The name of the object and files please refer to the example.cpp file.**  **Output:**  Please refer to the sample output  **Sample Input / Output：**   |  |  | | --- | --- | | Sample Input | Sample Output | | example.cpp | intList length : 10  you can not change this value  intList length : 10  intList index(2) : 0  the index is out of array  the index is out of array  doubleList index(2) : 0.48  the index is out of array  doubleList length : 0  the index is out of array  charList index(1) :  charList index(1) : b  alphabet : A  you can not change this value  integer : 50  you can not change this value | |
| **□ Easy, Only basic programming syntax and structure are required.**  **□ Medium, Multiple programming grammars, and structures are required.**  **■ Hard, Need to use multiple program structures or complex data types.** |
| **Expected solving time:**  40 minutes |
| **Other notes:**  The floating point values will be output according to default output settings namely you did not need to change the output settings.  All of exceptions:  Array<int> intList(10);  try {  intList.length = 20; // error. The message is “you can not change this value”.  }  catch (Exception e) {  std::cout << e.message() << std::endl;  }  try {  intList.set(20, 100); // error. The message is “the index is out of array”.  }  catch (Exception e) {  std::cout << e.message() << std::endl;  }  try {  std::cout << "intList index(20) : " << intList.get(20) << std::endl; // error. The message is “the index is out of array”.  }  catch (Exception e) {  std::cout << e.message() << std::endl;  }  intList.clear();  try {  std::cout << " intList index(2) : " << intList.get(2) << std::endl; // error. The message is “the index is out of array”.  }  catch (Exception e) {  std::cout << e.message() << std::endl;  } |