**CPP Problem Design Example**

|  |
| --- |
| **Subject: N Dim Vector** |
| **Contributor: 陳泳峰, 陳宥潤, 范茗翔** |
| **Main testing concept: Class Implementation**   |  |  | | --- | --- | | **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:**  Please define a class named **VecNf** to store a multidimensional vector and implement some simple vector operations.   * Class **VecNf** has three constructors: * **VecNf()** Construct a one-dimensional zero vector in default. * **VecNf(float \*dest, int cnt)** Construct a *cnt* dimensional vector using the first *cnt* numbers in *dest*. * **VecNf(const VecNf& rhs)** Construct a new vector from a VecNf. * Suppose there were two **VecNf**s: v1(2,2), v2(6,1). You are required to overload operations to meet the following operational requirements: * **Assignment:**   Assign the value from a **VecNf** to another polynomial. (define operator =)  For Example: *vector = v1*; Then *vector* is (2, 2).   * **Return the value** of the certain dimension variable in the vector.   (define operator [])  For example: *v1[0]* needs to return the value on dimension 0, which has the value of 2.   * **Addition:**   Implement the addition of two vectors. (define operator +)  For Example: *vector = v1 +* *v2*; Then *vector* is(8, 3).   * **Subtraction:**   Implement the subtraction of two vectors. (define operator )  For Example: *vector = v1 - v2*; Then *vector* is(-4, 1).   * **Multiplication:**   Implement the multiplication of two vectors. (define operator \*)  For Example: *vector = v1 \* v2*; Then *vector* is(12, 2).  \*\* Please print " ASSIGNMENT!!! " each time you call "=".  \*\* For any arithmetic operator, the two **VecNf** must have the same dimensional. If they are not, please print "dimensions inconsistent" and return a one-dimensional zero vector.  **Input:**  No inputs.  \*\* The main() function in your submission will be replaced when judging.  \*\* You can use the main() function in “Other Notes” to test your program.  **Output:**  The result of executing your program with the given main function.  **Sample Input / Output：**   |  |  | | --- | --- | | Sample Input | Sample Output | | No inputs. | ASSIGNMENT!!!  3 2  ASSIGNMENT!!!  1 2 3  ASSIGNMENT!!!  7 7 7  28  dimensions inconsistent  0 | |
| **■ 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:**  int main()  {  float a\_value[] = { 3.0, 2.0 };  float b\_value[] = { 1, 2, 3 };  float c\_value[] = { 6, 5, 4 };  VecNf A(a\_value, 2);  VecNf B(b\_value, 3);  VecNf C(c\_value, 3);  VecNf T;  T = A; // Assignment  for (int i = 0; i < T.Size(); i++) {  cout << T[i] << “ ”;  } cout << endl;  T = B; // Assignment  for (int i = 0; i < T.Size(); i++) {  cout << T[i] << “ ”;  } cout << endl;  T = B + C; // Vector addition  for (int i = 0; i < T.Size(); i++) {  cout << T[i] << “ ”;  } cout << endl;  cout << C \* B << endl; // Scale  cout << A \* C << endl; // Inconsistent    system("pause");  return 0;  } |