**CPP Problem Design Example**

|  |
| --- |
| **Subject：Template Binary Search** |
| **Contributor: 廖宣瑋,** **陳俊儒,** **林承達** |
| **Main testing concept: Templates**   |  |  | | --- | --- | | **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 implement Binary Search using template, and provide both iterative and recursive versions.  The iterative version of the function should follow this format:  **ItrBinarySearch(const T a[], int first , int last, T key, bool &found, int &location)**.  The recursive version of the function should follow this format:  **RecBinarySearch (const T a[], int first, int last, T key, bool &found, int &location)**.   * **a[]** is the list that will be searched. * **first** is the start position. * **last** is the end position. * **key** is the element to be searched. * **found** is for recording whether the **key** exists in the list **a[].** * **location** is the position of **key** in the list **a[]**.   \*\*Both iterative and recursive versions should support *int*, *string* and *double* types.  **Input:**  Please enter the number of times to be tested N, and then enter N sets of keys with type of int, string, and double, on a line by itself.  \*\*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 | | 3  1  aa  0.3  100  zk  2019.2  5  gg  2018.2 | Array contains:  1 2 3 4 10 25 29 100  Enter number to be located:  Testing Template Iterative Binary Search  1 is in index location 0  Testing Template Recursive Binary Search  1 is in index location 0  Array contains:  aa ab ah bd be cc fe zk  Enter number to be located:  Testing Template Iterative Binary Search  aa is in index location 0  Testing Template Recursive Binary Search  aa is in index location 0  Array contains:  0.3 5.6 7.8 10.9 123.5 150.1 197.1 2019.2  Enter number to be located:  Testing Template Iterative Binary Search  0.3 is in index location 0  Testing Template Recursive Binary Search  0.3 is in index location 0  Array contains:  1 2 3 4 10 25 29 100  Enter number to be located:  Testing Template Iterative Binary Search  100 is in index location 7  Testing Template Recursive Binary Search  100 is in index location 7  Array contains:  aa ab ah bd be cc fe zk  Enter number to be located:  Testing Template Iterative Binary Search  zk is in index location 7  Testing Template Recursive Binary Search  zk is in index location 7  Array contains:  0.3 5.6 7.8 10.9 123.5 150.1 197.1 2019.2  Enter number to be located:  Testing Template Iterative Binary Search  2019.2 is in index location 7  Testing Template Recursive Binary Search  2019.2 is in index location 7  Array contains:  1 2 3 4 10 25 29 100  Enter number to be located:  Testing Template Iterative Binary Search  5 is not in the array.  Testing Template Recursive Binary Search  5 is not in the array.  Array contains:  aa ab ah bd be cc fe zk  Enter number to be located:  Testing Template Iterative Binary Search  gg is not in the array.  Testing Template Recursive Binary Search  gg is not in the array.  Array contains:  0.3 5.6 7.8 10.9 123.5 150.1 197.1 2019.2  Enter number to be located:  Testing Template Iterative Binary Search  2018.2 is not in the array.  Testing Template Recursive Binary Search  2018.2 is not in the array. | |
| **□ 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 more complex data types.** |
| **Expected solving time:**  30 minutes |
| **Other notes:**  #include "Template.h"  int main(){  const int ARRAY\_SIZE = 8;  const int finalIndex = ARRAY\_SIZE - 1;  int count = 0;  cin >> count;  for (; count >= 1; count--){  int i;  int a[] = { 1, 2, 3, 4, 10, 25, 29, 100 };  // Test int  cout << "\nArray contains:\n";  for (i = 0; i < ARRAY\_SIZE; i++){  cout << a[i] << " ";  }  cout << endl;  int keyInt, location;  bool found;  cout << "Enter number to be located: ";  cin >> keyInt;  cout << "Testing Template Iterative Binary Search\n";  ItrBinarySearch(a, 0, finalIndex, keyInt, found, location);  if (found)  cout << keyInt << " is in index location " << location << endl;  else  cout << keyInt << " is not in the array." << endl;  cout << "Testing Template Recursive Binary Search\n";  RecBinarySearch(a, 0, finalIndex, keyInt, found, location);  if (found)  cout << keyInt << " is in index location " << location << endl;  else  cout << keyInt << " is not in the array." << endl;  // Test string  string b[] = {"aa", "ab", "ah", "bd", "be", "cc", "fe", "zk" };  string keyString;  cout << "\nArray contains:\n";  for (i = 0; i < ARRAY\_SIZE; i++){  cout << b[i] << " ";  }  cout << endl;  cout << "Enter number to be located: ";  cin >> keyString;  cout << "Testing Template Iterative Binary Search\n";  ItrBinarySearch(b, 0, finalIndex, keyString, found, location);  if (found)  cout << keyString << " is in index location " << location << endl;  else  cout << keyString << " is not in the array." << endl;  cout << "Testing Template Recursive Binary Search\n";  RecBinarySearch(b, 0, finalIndex, keyString, found, location);  if (found)  cout << keyString << " is in index location " << location << endl;  else  cout << keyString << " is not in the array." << endl;  // Test double  double c[] = { 0.3 , 5.6 , 7.8 , 10.9 , 123.5 , 150.1 , 197.1 , 2019.2 };  double keyDouble;  cout << "\nArray contains:\n";  for (i = 0; i < ARRAY\_SIZE; i++){  cout << c[i] << " ";  }  cout << endl;  cout << "Enter number to be located: ";  cin >> keyDouble;  cout << "Testing Template Iterative Binary Search\n";  ItrBinarySearch(c, 0, finalIndex, keyDouble, found, location);  if (found)  cout << keyDouble << " is in index location " << location << endl;  else  cout << keyDouble << " is not in the array." << endl;  cout << "Testing Template Recursive Binary Search\n";  RecBinarySearch(c, 0, finalIndex, keyDouble, found, location);  if (found)  cout << keyDouble << " is in index location " << location << endl;  else  cout << keyDouble << " is not in the array." << endl;  }  system("pause");  return 0;  } |