**CPP Problem Design**

|  |
| --- |
| **Subject: Form Word** |
| **Contributor: 謝公耀, 陳俊儒, 廖宣瑋** |
| **Main testing concept: File I/O**   |  |  | | --- | --- | | **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:**  Here is the rule of a word game: Give the player a word and the player should use the letters in the given word to compose new words. For example, with the given word "swimming", you can get words like "wing", "sing" and "miming".  Please design a class named "**Form**" to implement this word game. In the beginning, the system will give a certain word and the file name of a file that contains all the valid words. The program will read the file and find out all the valid words composed by the given word and print them.  You are required to implement following member functions:   * **void SetInputWord(string inputWord):** Set the given word of the game. * **void ProcessInputWord():** Some necessary pre-process of the given word. (e.g. upper or lower case transition.) * **void SetFileName(string fileName):** Set the file name of the file that contains all the valid words. * **void** **Load\_CompareWord():** Load the valid words from the file, and find out all the valid words composed by the letters in the given word. * **void PrintFoundWords():** Print the words found in the function Load\_CompareWord().   \*\*Note that the comparison is not case-sensitive, but the output should keep the case of the original letters.  **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:**  sample.out  **Sample Input / Output：**   |  |  | | --- | --- | | Sample Input | Sample Output | | No inputs. | g  gi  gim  gin  gins  gis  gm  gn  gns  gs  i  ign  ignis  ii  im  imi  immi  immis  in  ing  ins  is  ising  ism  isn  iw  iwis  m  mg  mi  mig  migs  mim  mimi  miming  min  ming  mini  minim  minims  minis  mins  mis  mm  mn  ms  msg  mw  n  ng  ni  nig  nim  nims  nis  nisi  nm  ns  s  sg  si  sig  sign  sim  sin  sing  sm  sn  snig  sw  swig  swim  swimming  swing  w  wg  wi  wig  wigs  wim  win  wing  wings  wins  wis  wising  wm  ws | |
| **□ Eazy,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:**  30 minutes |
| **Other notes:**  int main()  {  Form form1;  form1.SetInputWord("SWIMMING"); // set input  form1.ProcessInputWord(); // process input  form1.SetFileName("word.txt"); // set file name  form1.Load\_CompareWord(); // read-in data and compare  form1.PrintFoundWords(); // print answers  return 0;  } |