**Question #2 ( 32 points)**

Le fichier Question2.cpp est un pilote pour tester la fonction TriPartielBulles().

TriPartielBulles() est l’implantation de l’algorithme suivant :

|  |
| --- |
| **Niveau de travail**  Entrée : T[ ], un tableau de données  Debut et Fin, le début et la fin de l’intervalle de travail.  Sortie : T[]  Tâche : Balayer l’intervalle en poussant des « grands » éléments vers la fin de l’intervalle, en particulier, le plus grand élément sera poussé en position Fin.  TriPartielBubble( T[], Debut, Fin )  pour K = Debut jusqu’à (Fin-1)  si( T[K] > T[K+1] )  echanger( T[K], T[K+1] ) |

**Votre travail : Écrire quatre tests spécifiquement pour détecter les erreurs par un sur le contrôle de la boucle for qui se trouve dans la fonction TriPartielBulles().**

Chacun des tests a la forme suivante :

// Test #4 - détecter si le contrôle finit trop tard

cout << endl << "Test #4 - detecter si le controle finit trop tard."

<< endl << endl;

debut = 3;

fin = 7;

/\*

Proposez un test qui va permettre la détection que le contrôle

de l'intervalle commence trop tard.

\*/

// #0 1 2 3 4 5 6 7 8 9

long Tableau4[10] = { 7, 7, 7, 7, 7 ,7, 7, 7, 7, 7 };

// Original - sera modifié par l'appel à la fonction TriPartielBulles).

long Attendu4[10] = { 7, 7, 7, 7, 7 ,7, 7, 7, 7, 7 };

// Attendu (après l'appel à la fonction TriPartielBulles).

TriPartielBulles(Tableau4, debut, fin);

if (!sontEgauxDeuxTableaux(Tableau4, Attendu4, debut, fin))

{

// affichage d’information. Voir le fichier Question2.cpp.

}

Pour fins de présentation, les tableaux ont été remplis avec la valeur 7 dans toutes les positions et pour tous les tests. L’intervalle sous considération est l’intervalle [3, 7] dans les quatre cas.

Comme vous pouvez le voir, la plupart du code des quatre tests est déjà écrit. **Mais vous devez proposer ou modifier le contenu des tableaux** pour détecter les cas spécifiques mentionnés. C’est-à-dire qu’il faut remplacer les contenus des tableaux pour créer les cas tests.

**Vous avez 8 tableaux à modifier :**

**Tableau1 et Attendu1 pour le Test #1 - détecter si le contrôle commence trop tôt ;**

**Tableau2 et Attendu2 pour le Test #2 - détecter si le contrôle commence trop tard ;**

**Tableau3 et Attendu3 pour le Test #3 - détecter si le contrôle finit trop tôt ;**

**Tableau4 et Attendu4 pour le Test #4 - détecter si le contrôle finit trop tard.**

**Les quatre TableauX sont les contenus de départ de votre tests et les quatre AttenduX sont les contenus que les TableauX correspondants devraient avoir après l’appel à TriPartielBulles() si la fonction fait bien son travail.**

Pour vous aider à vérifier que vos tests détectent quelque chose, je vous ai fourni un moyen de mettre des erreurs dans le code. Il suffit de choisir quel « for » n’est pas en commentaires.

Pour la version « correcte ».

|  |
| --- |
| void TriPartielBulles(long T[], long Debut, long Fin)  {  /\*  Vous pouvez choisir le "bon bug" en enlevant les // devant l'un des cinq for  suivants tout en gardant un seul des for sans le // au début.  \*/  // for( long K = Debut-1 ; K < Fin ; K++ ) // Version avec erreur : commence trop tôt  // for( long K = Debut+1 ; K < Fin ; K++ ) // Version avec erreur : commence trop tard  // for( long K = Debut ; K < Fin-1 ; K++ ) // Version avec erreur : finit trop tôt  // for( long K = Debut ; K <= Fin ; K++ ) // Version avec erreur : finit trop tard  for (long K = Debut; K < Fin; K++) // Version correcte  if (T[K] > T[K + 1])  echanger(T[K], T[K + 1]);  } |

Version avec erreur : le contrôle commence trop tôt.

|  |
| --- |
| void TriPartielBulles(long T[], long Debut, long Fin)  {  /\*  Vous pouvez choisir le "bon bug" en enlevant les // devant l'un des cinq for  suivants tout en gardant un seul des for sans le // au début.  \*/  for( long K = Debut-1 ; K < Fin ; K++ ) // Version avec erreur : commence trop tôt  // for( long K = Debut+1 ; K < Fin ; K++ ) // Version avec erreur : commence trop tard  // for( long K = Debut ; K < Fin-1 ; K++ ) // Version avec erreur : finit trop tôt  // for( long K = Debut ; K <= Fin ; K++ ) // Version avec erreur : finit trop tard  // for (long K = Debut; K < Fin; K++) // Version correcte  if (T[K] > T[K + 1])  echanger(T[K], T[K + 1]);  } |

Version avec erreur : le contrôle commence trop tard.

|  |
| --- |
| void TriPartielBulles(long T[], long Debut, long Fin)  {  /\*  Vous pouvez choisir le "bon bug" en enlevant les // devant l'un des cinq for  suivants tout en gardant un seul des for sans le // au début.  \*/  // for( long K = Debut-1 ; K < Fin ; K++ ) // Version avec erreur : commence trop tôt  for( long K = Debut+1 ; K < Fin ; K++ ) // Version avec erreur : commence trop tard  // for( long K = Debut ; K < Fin-1 ; K++ ) // Version avec erreur : finit trop tôt  // for( long K = Debut ; K <= Fin ; K++ ) // Version avec erreur : finit trop tard  // for (long K = Debut; K < Fin; K++) // Version correcte  if (T[K] > T[K + 1])  echanger(T[K], T[K + 1]);  } |

Version avec erreur : le contrôle finit trop tôt.

|  |
| --- |
| void TriPartielBulles(long T[], long Debut, long Fin)  {  /\*  Vous pouvez choisir le "bon bug" en enlevant les // devant l'un des cinq for  suivants tout en gardant un seul des for sans le // au début.  \*/  // for( long K = Debut-1 ; K < Fin ; K++ ) // Version avec erreur : commence trop tôt  // for( long K = Debut+1 ; K < Fin ; K++ ) // Version avec erreur : commence trop tard  for( long K = Debut ; K < Fin-1 ; K++ ) // Version avec erreur : finit trop tôt  // for( long K = Debut ; K <= Fin ; K++ ) // Version avec erreur : finit trop tard  // for (long K = Debut; K < Fin; K++) // Version correcte  if (T[K] > T[K + 1])  echanger(T[K], T[K + 1]);  } |

Version avec erreur : le contrôle finit trop tard.

|  |
| --- |
| void TriPartielBulles(long T[], long Debut, long Fin)  {  /\*  Vous pouvez choisir le "bon bug" en enlevant les // devant l'un des cinq for  suivants tout en gardant un seul des for sans le // au début.  \*/  // for( long K = Debut-1 ; K < Fin ; K++ ) // Version avec erreur : commence trop tôt  // for( long K = Debut+1 ; K < Fin ; K++ ) // Version avec erreur : commence trop tard  // for( long K = Debut ; K < Fin-1 ; K++ ) // Version avec erreur : finit trop tôt  for( long K = Debut ; K <= Fin ; K++ ) // Version avec erreur : finit trop tard  // for (long K = Debut; K < Fin; K++) // Version correcte  if (T[K] > T[K + 1])  echanger(T[K], T[K + 1]);  } |

**Vous êtes évalués sur vos choix de tests. Est-ce que chacun de vos cas tests force la détection du cas en question.**

Vous trouverez dans le sous-répertoire Solution 5 fichiers Document texte :

* solution - erreur - commence trop tard.txt
* solution - erreur - commence trop tôt.txt
* solution - erreur - finit trop tard.txt
* solution - erreur - finit trop tôt.txt
* solution - version correcte.txt

Ces fichiers montrent ce à quoi pourrait ressembler vos sorties de programmes.

**Notez que j’ai édité les fichiers pour que tous les éléments des tableaux soient des 7 pour ne pas vous donnez mes exemples de cas tests.**

**N’allez pas écrire des cas tests tels que chacune des erreurs proposées déclenchent les quatre détections.**

**Vos cas tests devraient être tels qu’une erreur ne devrait déclencher de préférence qu’une des détections et au plus deux (sinon, c’est du n’importe quoi.)**

**Lors de la correction, je vais essayer votre programme avec les quatre erreurs et la version correcte pour voir comment vos cas tests se défendent.**

**Il y a 4 tests pour 32 points, donc 8 points par test.**