**Cas de test simple « 2 noms sur la même ligne dans le fichier facture »**

|  |  |
| --- | --- |
| **Cas de test** | Cas -1 |
| **Nom du cas de test** | Test « écrire 2 noms sur la même ligne dans le fichier facture » |
| **Auteur du cas de test** | Jérémie Bergeron |
| **Système ou sous-système** | Principal |
| **Une courte description du cas test** | Le test a pour but de voir comment réagit notre programme lorsqu’il y a 2 noms sur la même ligne dans la section « Client » qui est dans le fichier texte de la facture |
| **Date d’exécution** | 2020-04-16 |

|  |
| --- |
| **Les pré-conditions**  Fichier existant. Le fichier de facture *Bergeron\_Test\_integre\_1.txt* sera utilisé pour ce test. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Étapes** | **Actions** | **Réponses attendue** | **Succès/Échec** | **Commentaires** |
| **1** | Ouvrir un éditeur de texte |  |  |  |
| **2** | Écrire *Clients :* pour montrer au programme que la section des clients débute. Faire un retour-chariot. |  |  |  |
| **3** | Écrire le nom des clients en prenant soin de toujours faire un retour-chariot entre chacun des clients.  Dans ce test, ceci sera écrit :  Roger test  Céline  Steeve |  |  | **Comme vous pouvez le constater, le mot test à la droite du client Roger est un mot invalide. Il ne devrait pas être là.** |
| **4** | Écrire *Plats :* pour montrer au programme que la section des plats débute. Faire un retour-chariot. |  |  |  |
| **5** | Écrire le nom des plats et le prix du plat (Ex : nomPlat prix). Faire un retour-chariot entre chacun des plats.  Dans ce test, ceci sera écrit :  Poutine 10.5  Frites 2.5  Repas\_Poulet 15.75 |  |  |  |
| **6** | Écrire *Commandes :* pour montrer au programme que la section des commandes débute. Faire un retour-chariot. |  |  |  |
| **7** | Écrire le nom du client, le nom du repas commandé et la quantité commandé (Ex : nomClient nomPlat quantité). Faire un retour-chariot entre chacune des commandes.  Dans ce test, ceci sera écrit :  Roger Poutine 2  Roger Frites 1  Céline Frites 1 |  |  |  |
| **8** | Écrire *Fin* pour montrer au programme la fin du fichier. |  |  |  |
| **9** | Ouvrir un éditeur de texte et changer la ligne 9 du fichier principal.java pour «LireFichier fichier = new LireFichier("facture\_test\_integre\\Bergeron\_Test\_integre\_1.txt"); ». |  |  |  |
| **10** | Partir le programme. | Le fichier ne respecte pas le format demandé ! (ligne: 2) | Succès | Le programme a su relever l’erreur. |

**Cas de test simple « quantité trop élevé »**

|  |  |
| --- | --- |
| **Cas de test** | Cas -2 |
| **Nom du cas de test** | Test « quantité trop élevé » |
| **Auteur du cas de test** | Jérémie Bergeron |
| **Système ou sous-système** | Principal |
| **Une courte description du cas test** | Le test a pour but de voir comment réagit notre programme lorsque la quantité est énorme dans la section « commande » qui est dans le fichier texte de la facture |
| **Date d’exécution** | 2020-04-16 |

|  |
| --- |
| **Les pré-conditions**  Fichier existant. Le fichier de facture *Bergeron\_Test\_integre\_2.txt* sera utilisé pour ce test. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Étapes** | **Actions** | **Réponses attendue** | **Succès/Échec** | **Commentaires** |
| **1** | Ouvrir un éditeur de texte |  |  |  |
| **2** | Écrire *Clients :* pour montrer au programme que la section des clients débute. Faire un retour-chariot. |  |  |  |
| **3** | Écrire le nom des clients en prenant soin de toujours faire un retour-chariot entre chacun des clients.  Dans ce test, ceci sera écrit :  Simon  Jeremie  Anthony |  |  |  |
| **4** | Écrire *Plats :* pour montrer au programme que la section des plats débute. Faire un retour-chariot. |  |  |  |
| **5** | Écrire le nom des plats et le prix du plat (Ex : nomPlat prix). Faire un retour-chariot entre chacun des plats.  Dans ce test, ceci sera écrit :  Burger 7.50  Frites 3  Boisson\_Gazeuse 2.75 |  |  |  |
| **6** | Écrire *Commandes :* pour montrer au programme que la section des commandes débute. Faire un retour-chariot. |  |  |  |
| **7** | Écrire le nom du client, le nom du repas commandé et la quantité commandé (Ex : nomClient nomPlat quantité). Faire un retour-chariot entre chacune des commandes.  Dans ce test, ceci sera écrit :  Anthony Burger 4  Simon Burger 1  Jeremie Boisson\_Gazeuse 1  Simon Frites 200000000000000000000000000 |  |  | **Comme vous pouvez le constater, la quantité de la deuxième commande de Simon est extrêmement élevé. Elle est invalide.** |
| **8** | Écrire *Fin* pour montrer au programme la fin du fichier. |  |  |  |
| **9** | Ouvrir un éditeur de texte et changer la ligne 9 du fichier principal.java pour «LireFichier fichier = new LireFichier("facture\_test\_integre\\Bergeron\_Test\_integre\_10.txt"); ». |  |  |  |
| **10** | Partir le programme. | Erreur lors de la lecture de la quantité d'une commande. | Succès | Le programme a su relever l’erreur. |

**Cas de test simple**

|  |  |
| --- | --- |
| **Cas de test** | Cas -1 |
| **Nom du cas de test** | Test écrire une quantité négative de commande |
| **Auteur du cas de test** | Simon St-Germain |
| **Système ou sous-système** | Principal |
| **Une courte description du cas test** | Le test a pour but de voir comment réagit notre programme lorsqu’il y a une quantité négative decommande. |
| **Date d’exécution** | 2020-04-16 |

|  |
| --- |
| **Les pré-conditions**  Fichier existant. Le fichier de facture *St-Germain\_test\_integre\_1.txt* sera utilisé pour ce test. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Étapes** | **Actions** | **Réponses attendue** | **Succès/Échec** | **Commentaire** |
| **1** | Changer la ligne 10 du fichier principal.java pour «LireFichier fichier = new LireFichier("facture\\St-Germain\_test\_integre\_1.txt"); ». |  |  |  |
| **2** | Executer le programme. | Erreur, la commande 0 de Roger n'est pas valide, car la quantité commandé est -2. | **Succès** |  |

**Cas de test simple**

|  |  |
| --- | --- |
| **Cas de test** | Cas -2 |
| **Nom du cas de test** | Test écrire un nom erroné d'un plat dans "commande" |
| **Auteur du cas de test** | Simon St-Germain |
| **Système ou sous-système** | Principal |
| **Une courte description du cas test** | Le test a pour but de voir comment réagit notre programme lorsqu’il y a un plat avec un nom erroné dans la section commande. |
| **Date d’exécution** | 2020-04-16 |

|  |
| --- |
| **Les pré-conditions**  Fichier existant. Le fichier de facture *St-Germain\_test\_integre\_2.txt* sera utilisé pour ce test. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Étapes** | **Actions** | **Réponses attendue** | **Succès/Échec** | **Commentaire** |
| **1** | Changer la ligne 10 du fichier principal.java pour «LireFichier fichier = new LireFichier("facture\\St-Germain\_test\_integre\_2.txt"); ». |  |  |  |
| **2** | Executer le programme. | Erreur, il n'existe pas de plat nommé: Repaus\_Poulet. | **Succès** |  |

**Cas de test simple**

|  |  |
| --- | --- |
| **Cas de test** | Client non-existant dans la section client de la facture |
| **Nom du cas de test** | Gestion d’erreur 1 |
| **Auteur du cas de test** | Simon Delteil |
| **Système ou sous-système** | Principale |
| **Une courte description du cas test** | L’utilisateur a modifié la facture et l’affiche sur la console |
| **Date d’exécution** | 16 Avril 2020 |

|  |
| --- |
| **Les pré-conditions**  Fichier existant. Le fichier de facture DelteilSimon*\_Test\_Integre\_1.txt* sera utilisé pour ce test. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Étapes** | **Actions** | **Réponses attendue** | **Succès/Échec** | **Commentaire** |
| **1** | **Créer un nouveau fichier de facture appelé «DelteilSimon\_Test\_Integre\_1.txt» dans le dossier «facture\_test\_integre»** |  |  |  |
| **2** | **Écrire dans le fichier dans la section Clients**  **«Clients :**  **Roger**  **Céline**  **Steeve»** |  |  |  |
| **3** | **Écrire dans le fichier dans la section plats**  **«Plats :**  **Poutine 10.5**  **Frites 2.5**  **Repas\_Poulet 15.75»** |  |  |  |
| **4** | **Écrire dans le fichier dans la section commande**  **«Commande :**  **Roger Poutine 1**  **Céline Frites 1**  **Jack Repas\_Poulet 1**  **Fin»** |  |  |  |
| **5** | **Changer la ligne 10 du fichier principal.java pour «LireFichier fichier = new LireFichier(«facture\_test\_integre\\DelteilSimon\_Test\_Integre\_1»);** |  |  |  |
| **6** | **Lancer le programme** | Bienvenue chez Barette !  Factures :  Roger 12.07$  Céline 2.87$  Erreur, le client Jack n'existent pas. | **succès** |  |

**Cas de test simple**

|  |  |
| --- | --- |
| **Cas de test** | Commande d’un plat non-existant |
| **Nom du cas de test** | Gestion d’erreurs des factures 2 |
| **Auteur du cas de test** | Simon Delteil |
| **Système ou sous-système** | Principale |
| **Une courte description du cas test** | L’utilisateur a modifié la facture et l’affiche sur la console |
| **Date d’exécution** | 16 Avril 2020 |

|  |
| --- |
| **Les pré-conditions**  Fichier existant. Le fichier de facture DelteilSimon*\_Test\_Integre\_1.txt* sera utilisé pour ce test. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Étapes** | **Actions** | **Réponses attendue** | **Succès/Échec** | **Commentaire** |
| **1** | **Créer un nouveau fichier de facture appelé «DelteilSimon\_Test\_Integre\_2» dans le dossier «facture\_test\_integre»** |  |  |  |
| **2** | **Écrire dans ce fichier de facture dans la section clients**  **«Clients :**  **Roger**  **Céline**  **Steeve»** |  |  |  |
| **3** | **Écrire dans ce fichier de facture dans la section plats**  **«Plats :**  **Poutine 10.5**  **Frites 2.5**  **Repas\_Poulet 15.75»** |  |  |  |
| **4** | **Écrire dans ce fichier de facture dans la section commande**  **«Commandes :**  **Roger Poutine 1**  **Céline Frites 1**  **Steeve Hamburger 1**  **Fin»** |  |  |  |
| **5** | **Changer la ligne 10 du fichier principal.java pour «LireFichier fichier = new LireFichier(«facture\_test\_integre\\DelteilSimon\_Test\_Integre\_1»);** |  |  |  |
| **6** | **Lancer le programme** | Bienvenue chez Barette !  Factures :  Roger 12.07$  Céline 2.87$  Erreur, il n'existe pas de plat nommé: Hamburger. | **succès** |  |