Ensuite, il va effectuer une addition entre le chiffre 1 et le chiffre 2. Si la valeur obtenue est plus grande que la base des 2 valeurs, le programme va placer une retenue de 1 dans le tableau. Pour le résultat, si l'addition est plus grand que la base, il va retirer la base et placer cette valeur dans la variable de résultat.

Si la valeur obtenue est moins grande que la base, il va placer un 0 dans le tableau des retenues et placer la valeur dans la variable de résultat. Mais c'est la dernière opération, et que la retenue actuelle vaut 1, il va la placer en première position du string résultat.

4.1.11Fonction substractValue

Pour la fonction de soustraction, le principe reste le même que l'addition.

```
170     /// <summary>
171     /// Fonction permettant d'effectuer une soustraction en colone entre 2 valeurs de meme base entrée par l'utilisateur.
172     /// </summary>
173     /// <param name="value1">Valeur 1 de l'utilisateur convertie dans son format</param>
174     /// <param name="value2">Valeur 2 de l'utilisateur convertie dans son format</param>
175     /// <param name="baseValue">Prends la base de la valeur 1 ou 2 pour afficher le résultat en fonction</param>
176     /// <param name="result">Stock et permet l'affichage du résultat</param>
177     /// <param name="restraint">Tableau de int permettant de stocker les retenues lors de l'addition en colone. Est aussi utilisée lors de l'affichage</param>
178     private void substractValue(string value1, string value2, int baseValue, string result, int[] restraint)
179     {
180         char[] charValue1 = value1.ToCharArray();
181         restraint[value1.Length - 1] = 0;
182 
183         //pour chaque chiffre effectue une soustraction et place les retenues dans le tableau à l'index actuel
184         for (int i = value1.Length - 1; i >= 0; i--)
185         {
186             string stringValue1Unit = Convert.ToString(charValue1[i]);
187             string stringValue2Unit = Convert.ToString(value2[i]);
188 
189             int intValue1Unit = Convert.ToInt32(stringValue1Unit);
190             int intValue2Unit = Convert.ToInt32(stringValue2Unit);
```

Figure 20 Initialisation de la fonction sbstractValue

Une fois les valeurs transformée et envoyée dans la fonction, le programme va soustraire à chaque unité.

```
192             //si la soustraction est supérieur à 0 le programme ne va pas placer de retenues
193             if (intValue1Unit + restraint[i] - intValue2Unit >= 0)
194             {
195                 result = Convert.ToString(intValue1Unit + restraint[i] - intValue2Unit) + result;
196 
197             }
198             //sinon le programme diminue le chiffre suivant de 1 est place la base en retenue de la valeur actuelle
199             else
200             {
201                 string temp = Convert.ToString(value1[i - 1]);
202                 int valueDiminued = Convert.ToInt32(temp);
203                 temp = Convert.ToString(valueDiminued - 1);
204                 charValue1[i-1] = Convert.ToChar(temp);
205                 restraint[i] = baseValue;
206                 result = Convert.ToString(intValue1Unit + restraint[i]- intValue2Unit) + result;
207             }
208         }
209         showResult(value1, value2, result);
210     }
```

Figure 21 Système de soustraction et des retenues