# beCP
## 2022
# Tâche 2.1: Agents secrets (secretagents)

Author: Damien Leroy

Limite de temps: 2 s  Limite mémoire: 64 MB

*Note: Cette tâche est interactive. Veuillez consulter les instructions spéciales pour implémenter, compiler et tester votre programme. N'hésitez pas à demander de l'aide à un surveillant.*

Vos agents secrets ont été deployés sur le terrain. Afin d'annoncer leur présence aux autres, ils utilisent un système de communication qui annonce leur identifiant par onde radio. Pour éviter tout pistage, l'émetteur peut introduire des erreurs dans l'identifiant envoyé.

Votre tâche est d'écrire le code de l'encodeur qui convertit les noms d'agents en identifiants à 6 chiffres et du décodeur qui convertit les identifiants (possiblement altérés) en noms d'agent.

### Fonctions à implémenter

| C++ | `vector<int> encode(vector<string> agents)` |
|---|---|
| | Étant donnée une liste de longueur `N` avec des noms d'agents (tous différents), génère une liste d'identifiants pour ces agents. Cette fonction n'est appelée qu'une seule fois. |
| **return** | Une liste de longueur `N` contenant des identifiants (nombres à 6 chiffres) d'agents. L'identifiant à l'indice `i` correspond à l'agent `agents[i]`. |

| C++ | `string decode(vector<string> agents, int id)` |
|---|---|
| | Étant donnés une liste de longueur `N` avec les noms d'agents dans le même ordre que dans l'appel de `encode`, et un identifiant, renvoie le nom de l'agent qui correspond à cet identifiant. L'identifiant correspond toujours à un agent existant (avec une possible altération). Cette fonction est appelée au plus 60000 fois, toujours après `encode`. |
| **return** | Le nom d'agent qui correspond à `id`. |

### Limites générales

— $2 \leq N \leq 1000$

## Contraintes supplémentaires

| Sous-tâche | Points | Contraintes |
|---|---|---|
| A | 40 | $N \leq 100$, les identifiants à décoder ne sont **pas** altérés |
| B | 40 | $N \leq 100$, les identifiants à décoder ont, au plus, un chiffre modifié |
| C | 20 | les identifiants à décoder ont, au plus, un chiffre modifié |

## Détails techniques

Pour tester localement, nous vous fournissons un évaluateur (grader) (différent de celui qui tournera sur le serveur) que vous pouvez modifier librement. Ce fichier ne doit **pas** être soumis. Vous pouvez compiler localement votre code avec l'évaluation en faisant :

```
g++ -std=c++11 -Wall -Werror -Wshadow grader.cpp secretagents.cpp
```

## Exemple 1 (sous-tâche A : les identifiants ne sont pas altérés)

Supposons que `encode` est appelé avec la liste `agents` suivante :

| | | |
|---|---|---|
| Rohaan Wilkerson | Anwar Tierney | Caroline Scott |

Vous pourriez retourner, par exemple :

| | | |
|---|---|---|
| 1 (= 000001) | 900990 | 249234 |

Ensuite, la fonction `decode` peut être appelée plusieurs fois, avec la même liste `agents` que ci-dessus et avec les identifiants suivants :

| `id` | réponse attendue |
|---|---|
| 900990 | Anwar Tierney |
| 1 | Rohaan Wilkerson |

## Exemple 2 (sous-tâche B/C : les identifiants peuvent être altérés)

Supposons que `encode` est appelé avec la liste `agents` suivante :

| | | | |
|---|---|---|---|
| Murphy Aguirre | Kierran Heath | Anton Russo | Claudia Rogers |

Vous pourriez retourner, par exemple :

| | | | |
|---|---|---|---|
| 123456 | 990000 | 30 (= 000030) | 666666 |

Ensuite, la fonction `decode` peut être appelée plusieurs fois, avec la même liste `agents` que ci-dessus et avec les identifiants suivants :

| `id` | réponse attendue |
|---|---|
| 990400 | Kierran Heath |
| 800030 | Anton Russo |
| 0 (= 000000) | Anton Russo |
| 666666 | Claudia Rogers |

## Aide : Extraire des chiffres d'un nombre

Pour extraire un chiffre d'un nombre, vous devez utiliser les opérateurs division entière (/) et modulo (%). La division entière est une division dans laquelle la partie décimale est jettée, par exemple 1234/100 retourne l'entier 12. Le modulo retourne le reste de la division, donc 1234%100 retourne 34.

En combinant ces deux opérateurs, vous pouvez extraire un chiffre précis d'un nombre, par exemple (1234/100)%10 retourne 2, (1234/10)%10 retourne 3.

De la même manière, vous pouvez construire des nombres à partir de chiffres par multiplication. Si vous avez à répéter le nombre "12" deux fois, vous n'avez qu'à faire $12 * 100 + 12$. Si vous voulez ajouter 5 devant ce nombre, ajoutez $5 * 10000$ à ce nombre, etc.