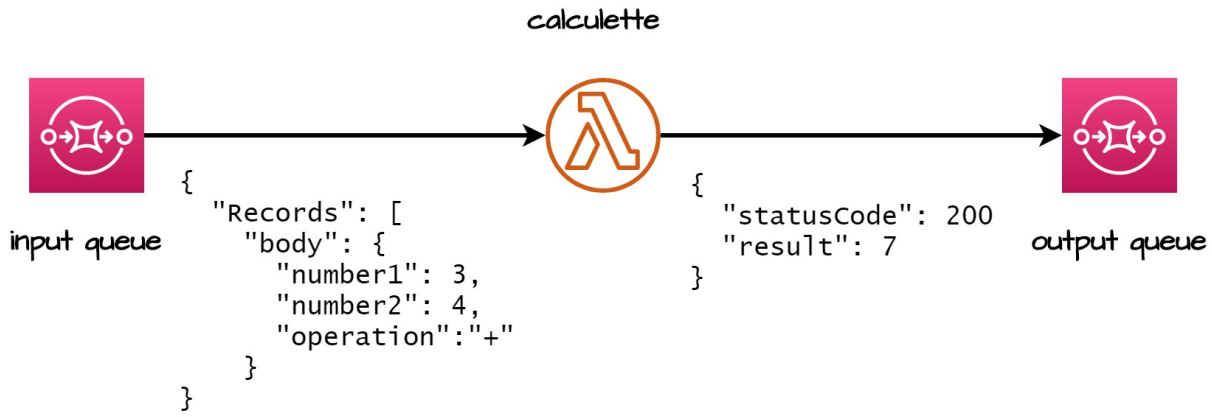
📦 Attendu

Cet exercice est à faire par groupe de 2 max. Vous pouvez ainsi le faire seul ou à deux. Vous noterez les membres du groupe dans un commentaire en entête du fichier `main.py`. Vous rendrez une Moodle une archive .zip contenant le code python de la fonction lambda et le code de l'infrastructure. **Attention votre code doit fonctionner tel quel**. C'est à dire qu'il suffit de faire un `cdktf deploy` pour tout lancer.

📜 Sujet

Dans cet exercice vous allez automatiser le déploiement du code final du TP3 via cdktf. Néanmoins vous allez devoir intégrer un léger changement à votre application. En effet il n'est pas possible avec cdktf de définir une cible pour une fonction lambda. À la place, vous allez poster les messages dans la queue SQS directement depuis le code de la lambda.



Pour faire cela vous allez utiliser le module `boto3` qui permet via python d'interagir avec les service AWS. Voici un code minimal pour poster dans une file SQS depuis python :

```python
import boto3
sqs = boto3.client('sqs')  #client is required to interact with
sqs.send_message(
   QueueUrl="https://sqs.us-east-1.amazonaws.com/XXXXXXX/YYYYYY", # Doit être remplacé la bonne url
   MessageBody="Hello world from boto3"
)
```

Pour résoudre cet exercice :

1. Créez via le terminal un projet cdktf . Créez un dossier `graded_lab` et ouvrez ce dossier dans VScode. Dans un terminal saisissez la commande `cdktf init --template="python" --providers="aws@~>4.0" --local` et validez les différentes questions (voir ce lien). Changez l'interpréteur python avec un `ctrl+shift+p` dans Vscode et chercher `Python: Slect Interpreter` et prenez celui qui pointe vers un virtualenv. Il a été crée pas Terraform et contient toutes les dépendances pour l'exercice (voir ce lien)

2. Commencez par créer les services nécessaires (les deux files, la lambda) en vous inspirant du cours. Déployez-les et vérifiez que tout se crée correctement avec la commande `cdktf deploy`
3. Récupérez l'url de la file qui vous servira d'output et ajoutez la comme variable environnement de votre lambda (cf cours). Vous pouvez accéder à l'url de votre file via son attribut `url` (cf TP2 quand vous alliez chercher l'id de certains services)
4. Mettez à jour le code de votre fonction lambda en utilisant le code ci-dessus et en récupérant l'url de votre file via `os.getenv()`. Faite que votre code boucle sur les messages s'il y en a plusieurs.
5. Redéployez votre application et testez-la.