



**POLYTECHNIQUE
MONTRÉAL**

UNIVERSITÉ
D'INGÉNIERIE

LOG3430 – MÉTHODES DE TEST ET DE VALIDATION DU LOGICIEL

TP3 : Fuzzing

Automne 2022

Groupe 3 (4) :

Victor Kim – 1954607

Jérémy Perreault – 1903274

Soumis à :

Maxime Lamothe

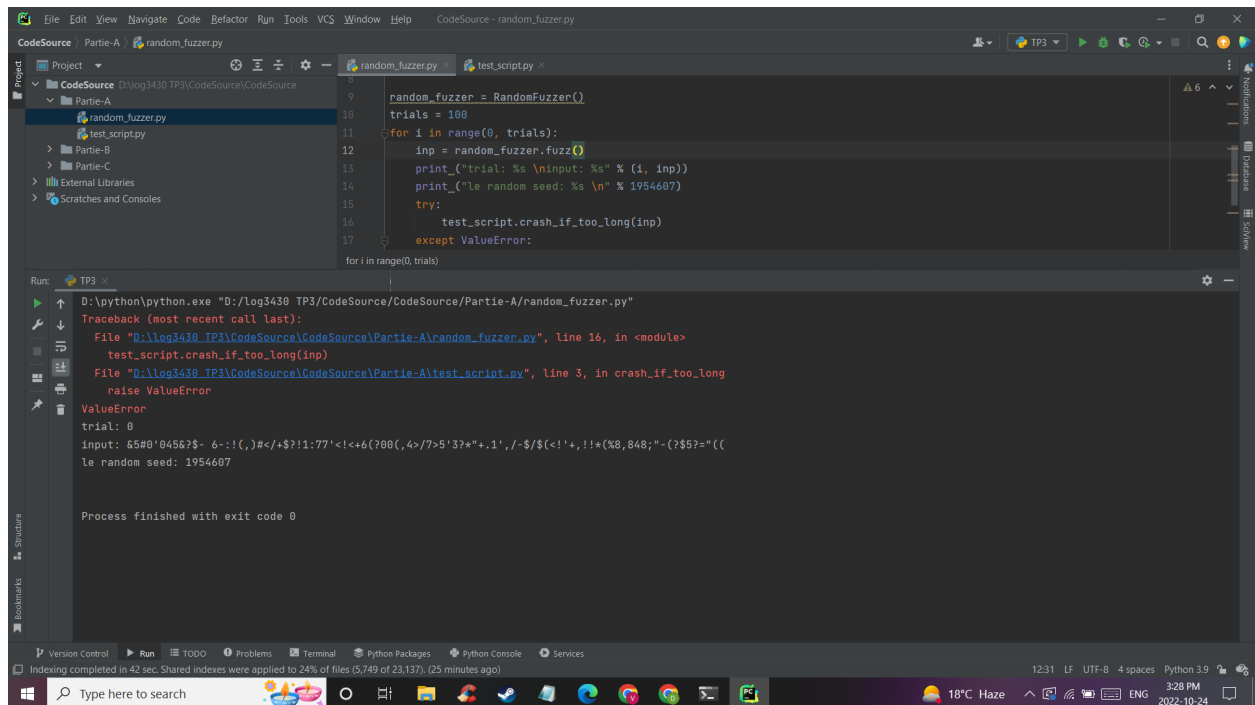
Dmytro Humenium

Adam Halim

[27 octobre 2022]

Question 1 : Collez la capture d'écran de la sortie de la partie A

La capture d'écran de la sortie de la Partie A est présentée à la Figure 1.



The screenshot shows a Python IDE with a project named 'Partie-A'. The code editor displays a file named 'random_fuzzer.py' with the following code:

```
9 random_fuzzer = RandomFuzzer()
10 trials = 100
11 for i in range(0, trials):
12     inp = random_fuzzer.fuzz()
13     print('trial: %s \ninput: %s' % (i, inp))
14     print('le random seed: %s \n' % 1954607)
15     try:
16         test_script.crash_if_too_long(inp)
17     except ValueError:
18         pass
19 for i in range(0, trials):
```

The Run console shows the execution output:

```
D:\python\python.exe "D:/log3430 TP3/CodeSource/CodeSource/Partie-A/random_fuzzer.py"
Traceback (most recent call last):
  File "D:/log3430 TP3/CodeSource/CodeSource/Partie-A/random_fuzzer.py", line 16, in <module>
    test_script.crash_if_too_long(inp)
  File "D:/log3430 TP3/CodeSource/CodeSource/Partie-A/test_script.py", line 3, in crash_if_too_long
    raise ValueError
ValueError
trial: 0
input: 65#0'045&?5- 6-!{(,)#</+$?1:77'<!<+6(700(,4>/7>5'3?+*.1',/-/$(<!'*,!!*(X8,848;"--(?5$?="(
le random seed: 1954607

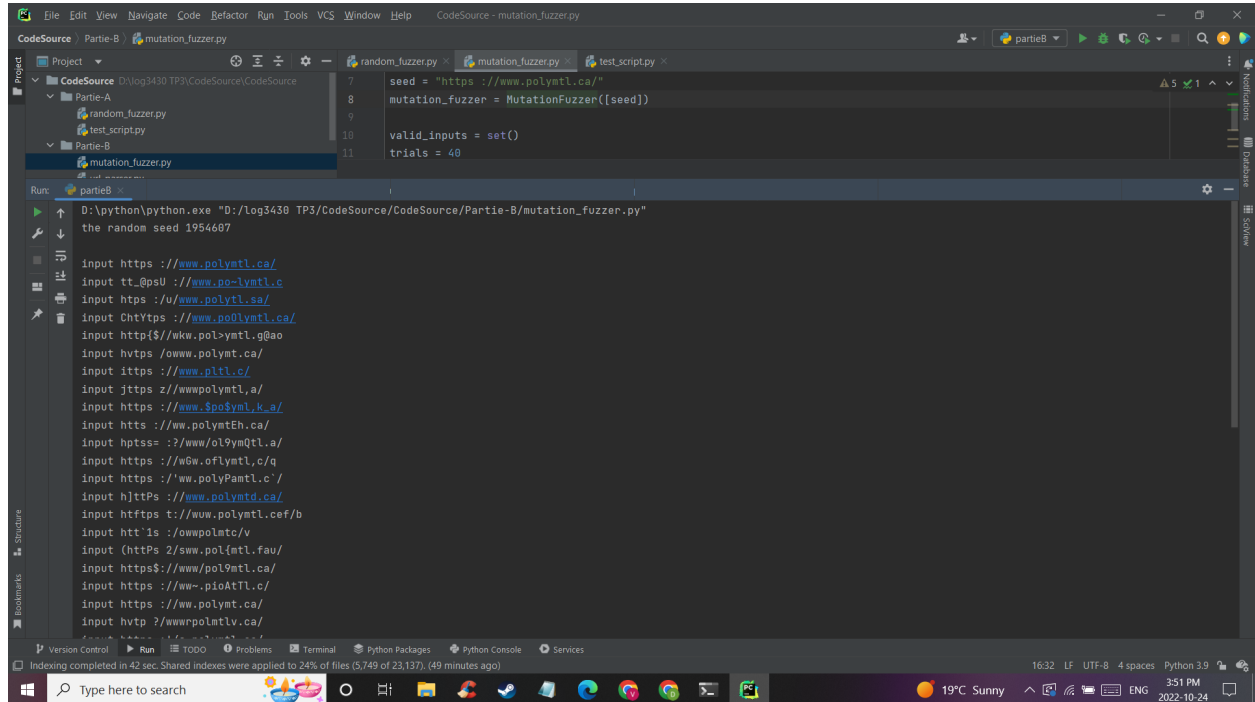
Process finished with exit code 0
```

Figure 1. Sortie de la partie A

Question 2 : Quel est le type de fuzzer dans la partie A ? Expliquez brièvement votre raisonnement.

Il s'agit d'un fuzzer de type BlackBox, puisqu'il ne prend pas en compte du fonctionnement du code à tester. En effet, il ne fait que générer une valeur aléatoire à partir de la semence qu'on lui fournit.

Question 3 : Collez la capture d'écran de la sortie de la partie B



The screenshot shows a VS Code editor with a project named 'CodeSource'. The file explorer on the left shows a directory structure with 'Partie-A', 'Partie-B', and 'mutation_fuzzer.py'. The editor window shows the code in 'mutation_fuzzer.py':

```
7 seed = "https://www.polytml.ca/"
8 mutation_fuzzer = MutationFuzzer([seed])
9
10 valid_inputs = set()
11 trials = 40
```

The Run console at the bottom shows the output of the script:

```
D:\python\python.exe "D:/log3430 TP3/CodeSource/CodeSource/Partie-B/mutation_fuzzer.py"
the random seed 1954607
input https://www.polytml.ca/
input tt@psu://www.po-lymtl.c
input https://u/www.polytml.ca/
input ChtYtps://www.po0lymtl.ca/
input http$//wkw.pol>ymtl.g@a0
input hvtps /owww.polytml.ca/
input ittps://www.pltl.c/
input jttps z//wwwpolymtl,a/
input https://www.$po$ymtl.k.a/
input https://ww.polytmEh.ca/
input hptss= :?/ww/al9ymQtl.a/
input https://w6w.oflymtl,c/q
input https://'ww.polyPamtl.c' /
input hJttPs://www.polymtl.ca/
input htftps t://www.polytml.cef/b
input htt'Is :/owwpolmtc/v
input (httPs 2/sww.pol(mtl.fau/
input https$://www/pol9mtl.ca/
input https://ww-.pioAtTL.c/
input https://www.polytml.ca/
input hvtp ?/wwrpolmtlv.ca/
```

Figure 2. Sortie de la partie B

Question 4 : Quel est le type de fuzzer dans la partie B? Expliquez brièvement votre raisonnement.

Il s'agit d'un fuzzer de type GreyBox, puisque le fonctionnement général de la méthode testée est connu, mais pas l'intégralité de son code. En effet, le fuzzer connaît le format de l'entrée, soit un URL, et teste toutes les variations/mutations qu'elle pourrait subir. Les tests ne sont donc pas complètement aléatoires; ils permettent de tester tous les cas limites.

Question 5 - a : Collez la capture d'écran du graphique que vous avez généré pour comparer les informations de couverture de Random Fuzzer, de Mutation Fuzzer et votre Fuzzer.

La capture d'écran du graphique généré lors de la partie C est présentée par la figure 3.

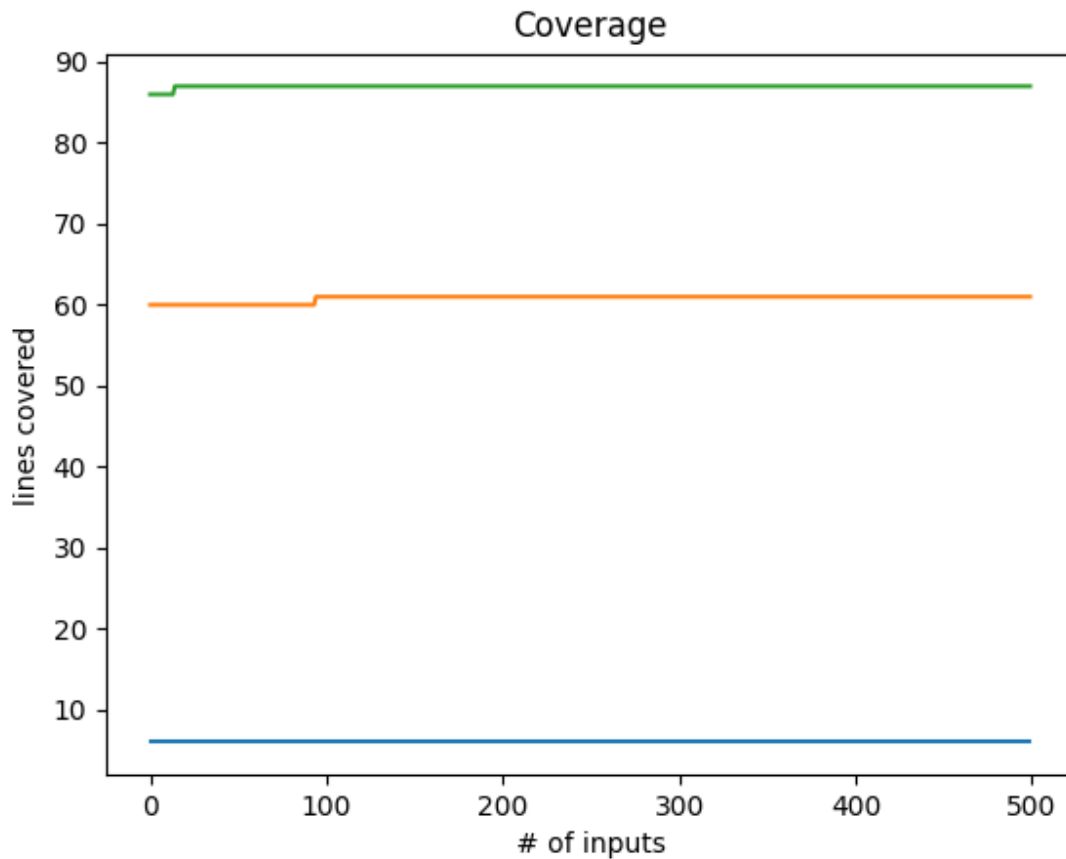


Figure 3. Graphique de la partie C

Question 5 - b : Décrivez le graphique

Le graphique représente le nombre de lignes couvertes en fonction du nombre d'entrées pour chaque Fuzzer. Il est possible de constater que le nombre de lignes couvertes pour le Random Fuzzer n'augmente pas avec le nombre d'entrées testées. Le graphique démontre que le Mutation Fuzzer permet d'obtenir une plus grande couverture que celle du Random Fuzzer et qu'elle est assez constante. Finalement, comme le graphique en témoigne, My Fuzzer fournit la meilleure couverture. Cela peut être expliqué par le fait qu'il a été implémenté en connaissant parfaitement le fonctionnement de num2words.

Question 5 - c : Incluez le code que vous avez implémenté en tant que q5.py (s'il s'agit d'un seul fichier) ou q5.zip (s'il y a plusieurs fichiers) avec votre rapport. Incluez la capture d'écran avec le code que vous avez ajouté, y compris l'implémentation du fuzzer, à votre rapport.

La capture d'écran de l'implémentation de notre Fuzzer est présentée par la figure 4. Le reste du code ajouté se trouve dans le fichier q5.py joint à ce rapport.

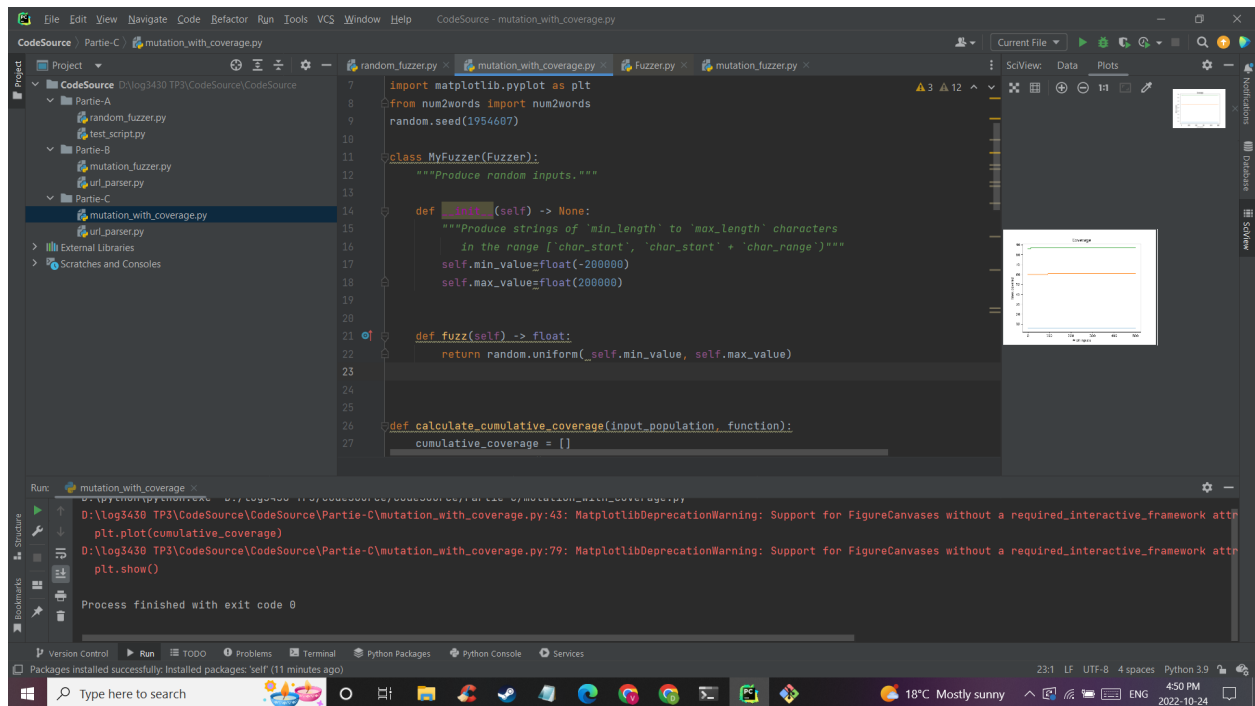
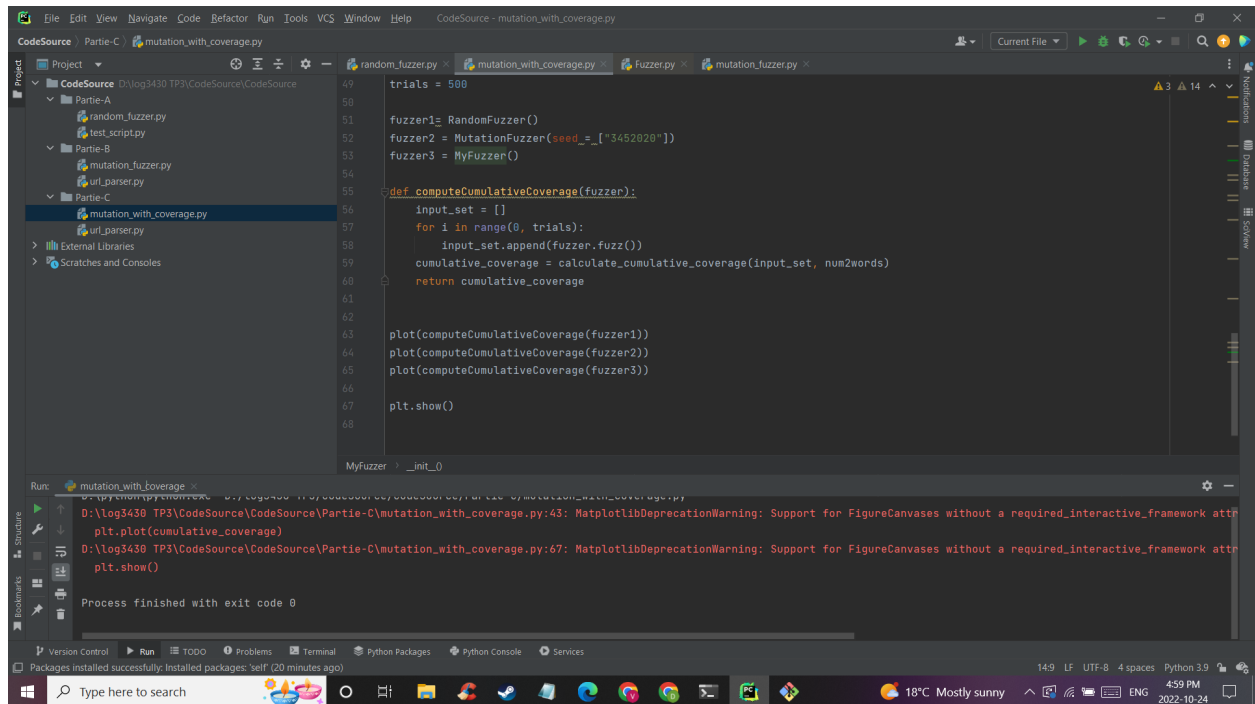


Figure 4. Implémentation de My Fuzzer



Question 6 : Quel est le type de fuzzer de mutation dans cette partie ? Expliquez brièvement votre raisonnement.

Le fuzzer de mutation dans cette partie est de type WhiteBox. Cela s'explique par le fait que nous connaissons le fonctionnement du Fuzzer étant donné que nous sommes les personnes qui l'ont développé. De plus, l'implémentation du Fuzzer a été réalisée en sachant l'implémentation du code à Tester. En effet, les sorties générées par le Fuzzer ont été conçues sur mesure pour tester le comportement de la méthode num2words. Ce qui explique que le nombre de lignes couvertes à la question 5 est plus grand pour ce fuzzer.