ΕΛΛΗΝΙΚΗ ΔΗΜΟΚΡΑΤΙΑ
Εθνικόν και Καποδιστριακόν
Πανεπιστήμιον Αθηνών
ΙΔΡΥΘΕΝ ΤΟ 1837

Εισαγωγή στον Προγραμματισμό

Εργασία #0

Οκτώβριος 2023

Στόχος της εργασίας είναι να αποκτήσουμε εμπειρία με τα βασικά εργαλεία του προγραμματιστή και να γράψουμε ένα από τα πρώτα προγράμματα σε C. Συγκεκριμένα, στοχεύουμε γνωριμία με τα ακόλουθα:

1. git, version control, repositories, GitHub
2. Γραμμή εντολών Linux
3. Χρήση ακεραίων στην C
4. Χρήση βασικών δομών (loops, if, functions)
5. Δεδομένα εισόδου μέσω ορισμάτων

Υποβολή Εργασίας. Όλες οι υποβολές των εργασιών θα γίνουν μέσω GitHub και συγκεκριμένα στο github.com/progintro [4]. Προκειμένου να ξεκινήσεις, μπορείς να δεχτείς την άσκηση με αυτήν την: πρόσκληση [5].

1 Νέο URL στο GitHub (20 Μονάδες)

Έχεις λογαριασμό στο GitHub; Έχεις ξαναφτιάξει repository; Έχεις φτιάξει URL στο διαδίκτυο; Μετά από αυτήν την άσκηση θα μπορείς να απαντήσεις "ναι" σε όλα. Αν ήδη μπορείς να απαντήσεις ναι, η άσκηση αυτή (ελπίζουμε) να μην σου πάρει πολύ χρόνο.

Η ιστοσελίδα του μαθήματος βασίζεται σε ένα public GitHub repository [2]. Σκοπός αυτής της άσκησης είναι να δημιουργήσεις ένα καινούριο repository στον προσωπικό σου λογαριασμό GitHub και να το συνδέσεις με GitHub Pages προκειμένου να είναι διαθέσιμο στο διαδίκτυο.

Τεχνικές Προδιαγραφές

Η υποβολή σου, πρέπει να έχει τα ακόλουθα χαρακτηριστικά:

- Repository Name: progintro/hw0-<YourUsername>
- Αρχείο Λύσης Filepath: pages/solution.txt
- README Filepath (optional): pages/README.md

Έστω ότι το GitHub username σου είναι YourUsername. Το αρχείο solution.txt πρέπει να περιέχει το URL με το domain YourUsername.github.io που δημιουργήσατε. Αν χρησιμοποιήσετε κάποιο *.github.io URL που δεν δημιουργήσατε η άσκηση μηδενίζεται. Ενδεικτική συμπεριφορά για μια επιτυχημένη υποβολή:

```
# Check out solution URL
$ cat solution.txt
YourUsername.github.io
# Ensure the URL exists
$ curl --output /dev/null --silent --head --fail YourUsername.github.io \
    && echo "URL exists" || echo "URL does not exist"
URL exists
```

Το αρχείο README.md είναι προαιρετικό αν θέλετε να προσθέσετε κάτι στην υποβολή σας.

2 Ξεκινώντας με την γραμμή εντολών (30 Μονάδες)

Έχεις χρησιμοποιήσει την γραμμή εντολών; Linux; Γνωρίζεις πως να κάνεις SSH σε απομακρυμένους servers και να λύνεις προβλήματα με εντολές κονσόλας; Αν όχι, καιρός να μάθουμε!

Προκειμένου να ξεκινήσεις, πήγαινε στην σελίδα: overthewire [3]. Για να ολοκληρώσεις την άσκηση, θα χρειαστεί να περάσεις όλα τα επίπεδα - μέχρι και το 10ο. Σε κάθε επίπεδο, απαιτείται να συνδεθείς μέσω SSH σε έναν διαφορετικό server με συγκεκριμένο port number και στην συνέχεια να λύσεις ένα μικρό πρόβλημα προκειμένου να αποκαλυφθεί ο κωδικός (ένα string μορφής Iequeiw2geecaeHee0losaa3aek) το οποίο είναι ο κωδικός SSH για το επόμενο επίπεδο.

Τεχνικές Προδιαγραφές

Η υποβολή σου, πρέπει να έχει τα ακόλουθα χαρακτηριστικά:

- Repository Name: progintro/hw0-<YourUsername>

- Αρχείο Λύσης Filepath: command/solution.txt
- README Filepath: command/README.md
- **Κάθε γραμμή που θα προστεθεί στο αρχείο solution.txt πρέπει να γίνει με διαφορετικό git commit. Κοινώς, θέλουμε να δούμε *12* διαφορετικά git commits. Η κάθε γραμμή πρέπει να προστεθεί όταν λύσετε το αντίστοιχο πρόβλημα, όχι όλες μαζί στο τέλος.**

Το περιεχόμενο του solution.txt πρέπει να έχει την ακόλουθη μορφή:

```
$ cat solution.txt
bandit0: bandit0
bandit1: ...
bandit2: ...
...
bandit10: ...
bandit11: ...
```

Όπου οι τελείες (...) περιγράφουν το μέρος της λύσης που έχει παραληφθεί. Μια σωστή λύση, θα έχει το ακόλουθο SHA hash [7]:

```
$ sha256sum solution.txt
a95444caa1805c4efa936a720d2761ee0f2f24af8347fbec7e6bd2834607a95c  solution.txt
```

Η λύση πρέπει να περιέχει όλους τους κωδικούς μέχρι και τον bandit11. Στο αρχείο README.md πρέπει να προσθέσετε λεπτομέρειες για τον τρόπο που λύσατε τα επίπεδα. Αν επιθυμείτε να λύσετε και τα επόμενα επίπεδα (φτάνουν μέχρι το 33! αλλά απαιτούν γνώσεις από μελλοντικά μαθήματα) προφανώς κάτι τέτοιο είναι ευπρόσδεκτο και λεπτομέρειες μπορούν να προστεθούν στο αρχείο README.md.

3 Η εικασία Collatz (50 Μονάδες)

Η εικασία Collatz είναι ένα από τα πιο διάσημα άλυτα προβλήματα των μαθηματικών [6]. Πολλοί μαθηματικοί έχουν επιχειρήσει κατά καιρούς να την αποδείξουν και μέχρι στιγμής δεν έχουμε ακόμα μια ευρέως αποδεκτή απόδειξη. Η εικασία ισχυρίζεται κάτι πολύ απλό: ότι η επανάληψη δυο απλών αριθμητικών πράξεων με μια συγκεκριμένη διαδικασία μπορεί να μετατρέψει οποιονδήποτε θετικό ακέραιο στο 1. Η διαδικασία των πράξεων έχει ως εξής:

1. Διάλεξε οποιονδήποτε θετικό ακέραιο N.
2. Αν ο αριθμός είναι 1, τότε τερμάτισε την διαδικασία.
3. Αν είναι άρτιος, ο επόμενος αριθμός στην ακολουθία θα είναι ο $N/2$.

4. Αν είναι περιττός, ο επόμενος αριθμός στην ακολουθία θα είναι ο $3 \times N + 1$.

5. Επανάλαβε τα βήματα 2-4 μέχρι να φτάσουμε στο 1.

Για παράδειγμα, έστω $N = 3$. Η παραπάνω διαδικασία θα παράξει την ακολουθία: 3, 10, 5, 16, 8, 4, 2, 1. Μπορείτε να δοκιμάσετε το ίδιο με τον αγαπημένο σας θετικό ακέραιο και να ελέγξετε την ακολουθία που παράγεται. Ο δικός μου για παράδειγμα είναι το 42 που οδηγεί στην ακολουθία: 42, 21, 64, 32, 16, 8, 4, 2, 1. Λογικά και η δική σας επιλογή κατέληξε στο 1 μετά από μερικά βήματα - αν όχι ίσως έχετε την ευκαιρία να γίνετε εκατομμυριούχοι [1] (ποιος είπε ότι τα μαθηματικά δεν έχουν λεφτά!).

Για κάθε θετικό ακέραιο N, ο αριθμός στοιχείων της ακολουθίας που παράγεται μέχρι να καταλήξουμε στο 1, λέγεται *μήκος ακολουθίας Collatz*. Για παράδειγμα, για $N = 3$, το μήκος της ακολουθίας είναι 8 (η ακολουθία έχει 8 στοιχεία: 3, 10, 5, 16, 8, 4, 2, 1). Αντίστοιχα για τον αριθμό 42, το μήκος ακουλουθίας Collatz είναι 9. Αν $N = 1$, τότε έχουμε το ελάχιστο μήκος 1.

Για το ζητούμενο αυτής της άσκησης, καλείστε να γράψετε ένα πρόγραμμα που βρίσκει το μέγιστο μήκος ακουλουθίας Collatz σε ένα εύρος αριθμών.

Τεχνικές Προδιαγραφές

- Repository Name: progintro/hw0-<YourUsername>
- Αρχείο C (Filepath): collatz/src/collatz.c
- Το αρχείο C που θα υποβληθεί πρέπει να μεταγλωττίζεται χωρίς ειδοποιήσεις για λάθη και με κωδικό επιστροφής (exit code) που να είναι 0. Συγκεκριμένα, το αρχείο σας **πρέπει** να μπορεί να μεταγλωττιστεί επιτυχώς με την ακόλουθη εντολή σε ένα από τα μηχανήματα του εργαστηρίου (linuxXY.di.uoa.gr):

  ```
  gcc -O0 -m32 -Wall -Wextra -Werror -pedantic -o collatz collatz.c
  ```

- README Filepath: collatz/README.md
- **Ένα αρχείο που να περιέχει ένα στοιχείο εισόδου και ένα εξόδου *διαφορετικά* από αυτά της άσκησης. Συγκεκριμένα προτείνουμε να βάλετε έναν συνδυασμό που θεωρείται ότι είναι δύσκολος να γίνει σωστός κατά την υλοποίηση**.
  - input Filepath: collatz/test/input.
  - output Filepath: collatz/test/output.

  Για παράδειγμα, το περιεχόμενο του input αρχείου μπορεί να είναι: "100 100000000" και του αντίστοιχου output: "950". Προσοχή: αυτό το παράδειγμα δεν θα γίνει δεκτό από την άσκηση επειδή αυτό το input-output ζευγάρι υπάρχει ήδη παραπάνω, επομένως πρέπει να διαλέξετε κάποιο άλλο.

- Όλοι οι ακέραιοι που θα δοθούν στο πρόγραμμά σας θα είναι στο εύρος

$$[-100000000, 100000000]$$

- Το ελάχιστο κάτω όριο που είναι δεκτό: 1.
- Το μέγιστο άνω όριο που είναι δεκτό: 100.000.000.
- Αν δοθεί οποισδήποτε μη θετικός αριθμός στα όρια η είσοδος θεωρείται μη αποδεκτή το πρόγραμμα πρέπει να τυπώνει: "0".
- Πρέπει να ολοκληρώνει την εκτέλεση μέσα σε: 4 λεπτά.
- Μέγιστο μέγεθος αρχείου C που μπορεί να χρησιμοποιηθεί στην υποβολή: 8KB.

Παρακάτω παραθέτουμε την αλληλεπίδραση με μια ενδεικτική λύση:

```
thanassis@linux14:~$ hostname
linux14
thanassis@linux14:~$ gcc -O0 -m32 -Wall -Wextra -Werror -pedantic -o collatz collatz.c
thanassis@linux14:~$ ./collatz 1 10
20
thanassis@linux14:~$ ./collatz 900 1000
174
thanassis@linux14:~$ ./collatz 80000 100000
333
thanassis@linux14:~$ ./collatz 100 1000000
525
thanassis@linux14:~$ ./collatz 100 100000000
950
thanassis@linux14:~$ ./collatz -1 100
0
```

Ενδεικτικοί χρόνοι:

```
thanassis@linux14:~$ time ./collatz 100 1000000
525

real 0m1,343s
user 0m1,338s
sys 0m0,004s
thanassis@linux14:~$ time ./collatz 100 100000000
950

real 3m0,212s
user 3m0,206s
sys 0m0,000s
```

Ο παραπάνω χρόνος μπορεί να είναι και πιο γρήγορος ανάλογα με την υλοποίηση / μεταγλώττιση / μηχάνημα στο οποίο τρέχει. Αν επιθυμείτε να τα βελτιώσετε ίσως χρειαστεί λίγο παραπάνω έρευνα από μέρους σας σε θέματα για τα οποία δεν έχουμε μιλήσει ακόμα στο μάθημα, αλλά φυσικά πιο γρήγορες υποβολές είναι ευπρόσδεκτες:

```
thanassis@linux14:~$ time ./collatz_fast 100 100000000
950

real 0m8,034s
user 0m6,789s
sys 0m1,244s
```

Στο αρχείο README.md πρέπει να προσθέσετε οποιεσδήποτε παρατηρήσεις σας κατά την διεκπεραίωση της άσκησης. Ο κώδικας απαιτείται να είναι καλά τεκμηριωμένος με σχόλια καθώς αυτό θα είναι μέρος της βαθμολόγησης. Για την υποβολή με την καλύτερη απόδοση (πιο γρήγορη από πλευράς χρόνου) θα ζητήσουμε μια γρήγορη (5λεπτη) παρουσίαση στο μάθημα, για την οποία θα λάβει +100% της βαθμολογίας (+50 Μονάδες). Σε περίπτωση ισοβαθμιών, θα ελέγξουμε την τεκμηρίωση των εργασιών. Αν δεν καταστεί δυνατό να λύσουμε τις ισοβαθμίες, μπορεί να επιλέξουμε παραπάνω από μια υποβολή.

Αναφορές

[1] https://www.prnewswire.com/news-releases/bakuage-offers-prize-of-120-million-jpy-to-whoever-solves-collatz-conjecture-math-problem-unsolved-for-84-years-301326629.html.

[2] Repository για το μάθημα . https://github.com/progintro/progintro.github.io.

[3] Ασκήσεις σε Γραμμή Εντολών . https://overthewire.org/wargames/bandit/bandit0.html.

[4] Οργανισμός για το μάθημα (GitHub progintro) . https://github.com/progintro.

[5] Πρόσκληση για Εργασία 0 . https://classroom.github.com/a/T_j7GPKK.

[6] Collatz Conjecture. https://en.wikipedia.org/wiki/Collatz_conjecture.

[7] SHA Hashing. https://en.wikipedia.org/wiki/SHA-2.