ΕΛΛΗΝΙΚΗ ΔΗΜΟΚΡΑΤΙΑ
Εθνικόν και Καποδιστριακόν Πανεπιστήμιον Αθηνών
ΙΔΡΥΘΕΝ ΤΟ 1837

Εισαγωγή στον Προγραμματισμό

Εργασία #1

Νοέμβριος 2025

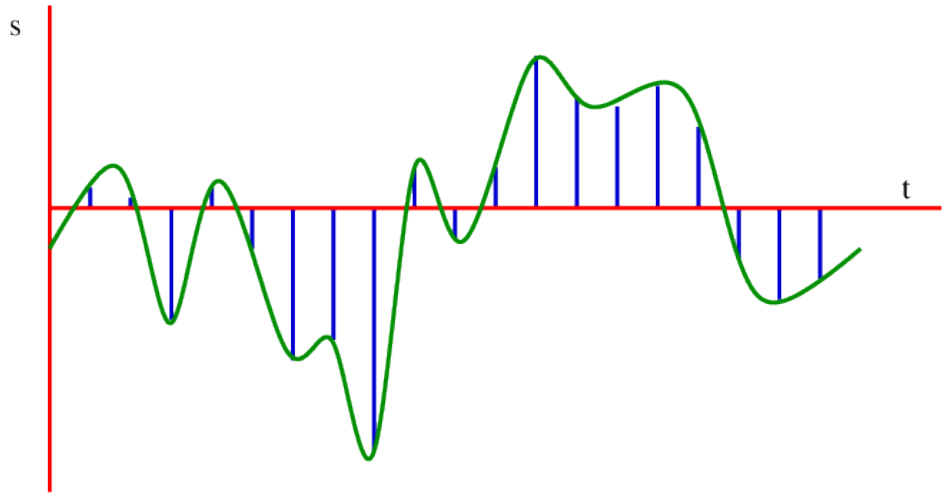
Στόχος αυτής της εργασίας είναι να γράψουμε ένα μεγαλύτερο πρόγραμμα σε γλώσσα C με χρήση βασικών τύπων και συναρτήσεων. Συγκεκριμένα, στοχεύουμε σε εμπειρία με τα ακόλουθα:

1. Χρήση ορισμάτων από την γραμμή εντολών
2. Χρήση συναρτήσεων (δικών μας αλλά και από την math.h)
3. Χρήση stdin / stdout / stderr
4. Χρήση δομών επανάληψης
5. Ανάγνωση και μετατροπή δυαδικών αρχείων

Υποβολή Εργασίας. Όλες οι υποβολές των εργασιών θα γίνουν μέσω GitHub και συγκεκριμένα στο github.com/progintro [1]. Προκειμένου να ξεκινήσεις, μπορείς να δεχτείς την άσκηση με αυτήν την: πρόσκληση [2].

1. Επεξεργασία Ήχου - soundwave (360 Μονάδες)

Ο ήχος είναι ένα αναλογικό σήμα, δηλαδή, από μαθηματική σκοπιά, μπορεί να θεωρηθεί σαν μία συνάρτηση f(t) ενός μεγέθους (της έντασης του ήχου) ως προς τον χρόνο. Το μέγεθος αυτό μπορεί να είναι η πυκνότητα του αέρα που διεγείρει το τύμπανο ενός αυτιού, το ηλεκτρικό σήμα που παράγεται από την κεφαλή ενός πικάπ όταν αυτή διατρέχει ένα αυλάκι ενός δίσκου βινυλίου, κλπ. Όμως, για να αποθηκευθεί ο ήχος σε έναν υπολογιστή, αυτό δεν μπορεί να γίνει στην αναλογική του μορφή, αλλά πρέπει να ψηφιοποιηθεί. Η ψηφιοποίηση αυτή γίνεται μέσω της καταγραφής της τιμής της συνάρτησης του σήματος σε διακριτές ισαπέχουσες χρονικές στιγμές, όπως φαίνεται στο Σχήμα 1. Οι τιμές της συνάρτησης μπορεί να θεωρηθεί ότι αντιστοιχούν στην ένταση του ήχου στις αντίστοιχες χρονικές στιγμές.



Σχήμα 1: Ο ήχος είναι ένα κύμα που ταξιδεύει στον χωροχρόνο. Τυπικά το φανταζόμαστε ως μια μαθηματική συνάρτηση που εκφράζει την ένταση του ήχου κάθε χρονική στιγμή αλλά υπάρχουν και εναλλακτικοί τρόποι σκέψης [8].

Αν καταγραφή των διαδοχικών τιμών της συνάρτησης είναι αρκετά πυκνή, παρότι δεν έχουμε πλέον στη διάθεσή μας το αρχικό αναλογικό σήμα, η αναπαραγωγή του ήχου από την ψηφιοποιημένη μορφή του μπορεί να είναι επαρκώς πιστή στην αρχική αναλογική εκδοχή του. Για παράδειγμα, η τυπική συχνότητα δειγματοληψίας σε ένα μουσικό CD είναι 44.1kHz, δηλαδή έχουν καταγραφεί 44.100 δεδομένα για κάθε δευτερόλεπτο μουσικής—επιλογή της Sony από το 1979 [6, 5].

Υπάρχουν διάφορα πρότυπα για την αποθήκευση ψηφιοποιημένων ήχων σε αρχεία, όπως, για παράδειγμα, wav, mp3/4, aac, m4a, wma, κλπ. Αντικείμενο αυτής της εργασίας είναι η δημιουργία *ενός συστήματος επεξεργασίας ήχου για αρχεία τύπου wav*. Πριν προχωρήσουμε στις τεχνικές προδιαγραφές της εργασίας, παραθέτουμε μια περιγραφή της μορφοποίησης (format) των αρχείων wav που θα πρέπει να διαχειριστούμε στα πλαίσια της εργασίας. Αν σας ενδιαφέρει, μπορείτε να βρείτε αναλυτικότερες και πληρέστερες περιγραφές για κάθε πεδίο online [7, 3].

Τα αρχεία wav είναι δυαδικά—δηλαδή τα περιεχόμενά τους δεν είναι άμεσα αναγνώσιμα ως κείμενο αλλά αντιθέτως περιέχουν μια σειρά από bytes διατεταγμένα σε ένα αρχείο στον δίσκο με συγκεκριμένες ερμηνείες. Τα αρχεία wav που θα κληθούμε να διαχειριστούμε στα πλαίσια της εργασίας έχουν την ακόλουθη δομή:

FileHeader | SampleData | OtherData

Δηλαδή αποτελούνται από μια κεφαλίδα (FileHeader) που περιέχει πληροφορίες για το συγκεκριμένο αρχείο ήχου wav, ακολουθούμενο από τα ίδια τα δείγματα του ήχου (SampleData) και στην συνέχεια ακολουθούν προαιρετικά άλλα δεδομένα (OtherData). Τα αρχεία κεφαλίδας wav, περιέχουν τα ακόλουθα πεδία με την σειρά:

- 4 bytes με τους χαρακτήρες RIFF (αρχικά του Resource Interchange File Format, που είναι το πρότυπο που χρησιμοποιείται για τη δομή των wav αρχείων).

- 4 bytes, συμβολικά *SizeOfFile*, για το μέγεθος του αρχείου, που στην πραγματικότητα είναι το πλήθος των bytes που ακολουθούν μέχρι το τέλος του αρχείου. Ουσιαστικά, είναι το μέγεθος του αρχείου στον δίσκο μείον 8 bytes, δηλαδή αυτά που καταλαμβάνουν το τρέχον πεδίο και το προηγούμενο (RIFF). Σημειώνεται ότι η αναπαράσταση ακεραίων χωρίς πρόσημο γίνεται κατά σειρά από το λιγότερο σημαντικό byte προς το περισσότερο σημαντικό (πρότυπο little-endian [4]). Δηλαδή, ο ακέραιος 0xA3870FB9 αναπαρίσταται κατά σειρά με τα bytes 0xB9, 0x0F, 0x87, 0xA3.

- 4 bytes με τους χαρακτήρες WAVE, που υποδεικνύουν τη συγκεκριμένη υποκατηγορία αρχείων που ακολουθούν το πρότυπο RIFF.

- 4 bytes με τους χαρακτήρες "fmt " (προσοχή στον τελευταίο χαρακτήρα κενού διαστήματος), που υποδεικνύουν την έναρξη του τμήματος format (format chunk) του αρχείου.

- 4 bytes για τον χώρο που καταλαμβάνουν τα δεδομένα του τμήματος format που θα ακολουθήσουν. Στα αρχεία που θα χρησιμοποιηθούν στην εργασία, η τιμή αυτού του πεδίου θα είναι πάντα 16 (0x00000010). Και εδώ εφαρμόζεται το πρότυπο little-endian, δηλαδή ο αριθμός αυτός αποθηκεύεται με την ακολουθία από bytes 0x10, 0x00, 0x00, 0x00.

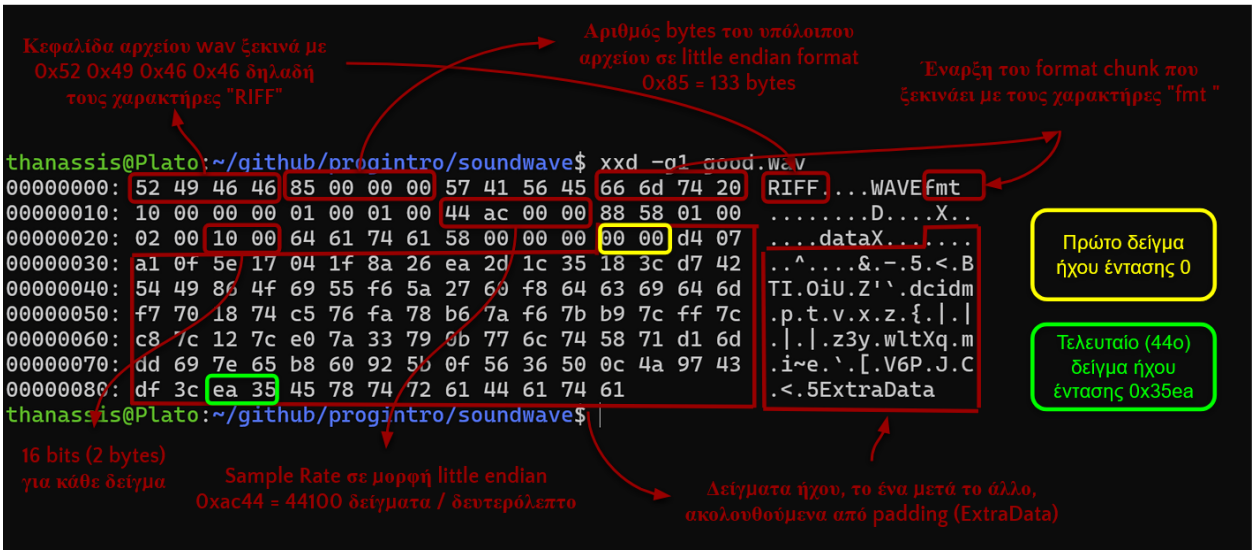
- 2 bytes για τον τύπο του WAVE format. Για την εργασία, αυτός θα ισούται πάντα με 1 (0x0001) και θα αποθηκεύεται κατά little-endian, δηλαδή με τα bytes 0x01, 0x00.

- 2 bytes, συμβολικά *MonoStereo*, για το αν ο ήχος είναι μονοφωνικός (τιμή 0x0001) ή στερεοφωνικός με δύο κανάλια (τιμή 0x0002). Προσοχή στο little-endian, όπως και σε όλες τις καταχωρήσεις ακεραίων που ακολουθούν.

- 4 bytes, συμβολικά *SampleRate*, για το ρυθμό δειγματοληψίας, δηλαδή πόσες τιμές, ανά δευτερόλεπτο, της συνάρτησης του αναλογικού ήχου έχουν καταγραφεί στο αρχείο.

- 4 bytes, συμβολικά *BytesPerSec*, για το πλήθος bytes ανά δευτερόλεπτο ήχου, που είναι καταχωρημένα στο αρχείο.

- 2 bytes, συμβολικά *BlockAlign*, για το πλήθος των bytes που απαιτούνται για την καταχώρηση της πληροφορίας του ήχου σε μία χρονική στιγμή, για όλα τα κανάλια. Σημειώστε ότι πάντοτε θα πρέπει να ισχύει ότι *BytesPerSec* = *SampleRate* $\times$ *BlockAlign*.



Σχήμα 2: Τα bytes ενός αρχείου wav με έμφαση σε κάποια από τα πεδία που είναι απαραίτητα για την αναπαράστασή του.

- 2 bytes, συμβολικά *BitsPerSample*, για το πλήθος των bits που απαιτούνται για την καταχώρηση της πληροφορίας του ήχου σε μία χρονική στιγμή, για ένα μόνο κανάλι. Στα πλαίσια της εργασίας, αυτή η τιμή θα είναι είτε 8 (0x0008), είτε 16 (0x0010). Για *BitsPerSample* = 8, η ένταση του ήχου είναι ένας ακέραιος χωρίς πρόσημο στο διάστημα [0,255], ενώ για *BitsPerSample* = 16, η ένταση του ήχου είναι ένας ακέραιος με πρόσημο στο διάστημα [-32768,32767]. Σημειώστε ότι πάντοτε θα πρέπει να ισχύει ότι *BlockAlign* = *BitsPerSample*/8 × *MonoStereo*.

- 4 bytes με τους χαρακτήρες data, που υποδεικνύουν την έναρξη του τμήματος data (data chunk) του αρχείου.

- 4 bytes, συμβολικά *SizeOfData*, για τον χώρο που καταλαμβάνουν τα δεδομένα του τμήματος data που θα ακολουθήσουν.

Μετά την κεφαλίδα του αρχείου, ακολουθούν τα ψηφιοποιημένα δεδομένα του ήχου με χρονική σειρά (SampleData). Στην περίπτωση στερεοφωνικού ήχου με δύο κανάλια, για κάθε χρονική στιγμή, υπάρχει ένα ζευγάρι δεδομένων, με το πρώτο στοιχείο να αντιστοιχεί στο αριστερό κανάλι και το δεύτερο στοιχείο στο δεξιό κανάλι.

Μετά το τέλος των δεδομένων ήχου, ενδεχομένως να υπάρχουν και άλλα τμήματα (OtherData) που προβλέπονται για την κατηγορία WAVE, με τα οποία όμως δεν θα ασχοληθούμε στην εργασία.

Ας δούμε τώρα τα περιεχόμενα ενός πραγματικού αρχείου wav. Μπορούμε να δούμε τα δυαδικά περιεχόμενα οποιουδήποτε αρχείου χρησιμοποιώντας την εντολή `od` ως εξής (στο Σχήμα 2 μπορείτε επίσης να δείτε μερικές εξηγήσεις για τα διάφορα πεδία σε συνδυασμό με την εντολή xxd):

```
$ od -tx1 good.wav
0000000 52 49 46 46 85 00 00 00 57 41 56 45 66 6d 74 20
0000020 10 00 00 00 01 00 01 00 44 ac 00 00 88 58 01 00
0000040 02 00 10 00 64 61 74 61 58 00 00 00 00 00 d4 07
0000060 a1 0f 5e 17 04 1f 8a 26 ea 2d 1c 35 18 3c d7 42
0000100 54 49 86 4f 69 55 f6 5a 27 60 f8 64 63 69 64 6d
0000120 f7 70 18 74 c5 76 fa 78 b6 7a f6 7b b9 7c ff 7c
0000140 c8 7c 12 7c e0 7a 33 79 0b 77 6c 74 58 71 d1 6d
0000160 dd 69 7e 65 b8 60 92 5b 0f 56 36 50 0c 4a 97 43
0000200 df 3c ea 35 45 78 74 72 61 44 61 74 61
0000215
```

Σημείωση

Πέρα από την εντολή od, υπάρχουν και άλλες εντολές όπως οι: (1) xxd, (2) hexdump, (3) hexedit που προσφέρουν αντίστοιχες δυνατότητες σε Unix-like συστήματα (Linux, MacOS, WSL κοκ). Για να *συγκρίνετε* δύο δυαδικά αρχεία, μπορείτε να χρησιμοποιήσετε την εντολή cmp ή την εντολή vbindiff.
Για περιβάλλοντα Windows, υπάρχουν ελεύθερα διαθέσιμα προγράμματα που λειτουργούν παρόμοια (π.χ. https://mh-nexus.de/en/hxd/).

Φτάσαμε λοιπόν στο ζητούμενο αυτής της άσκησης: να υλοποιήσουμε ένα πρόγραμμα επεξεργασίας αρχείων wav με δυνατότητες αλλαγής και παραγωγής ήχου—όλα σε γλώσσα C! Οι τεχνικές προδιαγραφές της υλοποίησης ακολουθούν.

Τεχνικές Προδιαγραφές

- Repository Name: progintro/hw1-<YourUsername>
- README Filepath: soundwave/README.md
- Αρχείο C (Filepath): soundwave/src/soundwave.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 -Ofast -Wall -Wextra -Werror -pedantic -o soundwave soundwave.c -lm

- Δύο αρχεία που να περιέχουν στοιχεία εισόδου και εξόδου διαφορετικά από αυτά της άσκησης για την εντολή `./soundwave rate 2.0`. Συγκεκριμένα προτείνουμε να βάλετε έναν συνδυασμό που θεωρείται ότι είναι δύσκολος να γίνει σωστός.

  - input Filepath: soundwave/test/input.wav
  - output Filepath: soundwave/test/output.wav

- Όλα τα περιεχόμενα των αρχείων wav θα δίνονται στο πρόγραμμά σας μέσω της πρότυπης εισόδου (stdin) και πρέπει να διαβάζονται με χρήση της συνάρτησης getchar. Δεν θα γίνουν δεκτές υλοποιήσεις όπου χρησιμοποιήθηκε κάποια άλλη συνάρτηση εισόδου εκτός της getchar.

- Όλες οι υποεντολές σας πρέπει να επιστρέφουν κωδικό εξόδου (exit code) 0 όταν ολοκλήρωσαν την επεξεργασία τους επιτυχώς - διαφορετικά πρέπει να επιστρέφουν με κωδικό εξόδου 1.

- Το πρόγραμμά σας θα πρέπει να υλοποιεί τις ακόλουθες 5 υποεντολές:

  - `info`: για τύπωμα των στοιχείων του wav και έλεγχο ορθότητας.
  - `rate`: για αλλαγή της ταχύτητας αναπαραγωγής (αντίστοιχο με το playback speed σε streaming υπηρεσίες).
  - `channel`: για επιλογή του καναλιού που θέλουμε.
  - `volume`: για μετατροπή της έντασης του ήχου.
  - `generate`: για παραγωγή ήχου.

  Αναλυτικές προδιαγραφές για την κάθε υποεντολή δίνονται στην συνέχεια.

- **Περιορισμοί Μνήμης**. Το πρόγραμμά σας πρέπει να χρησιμοποιεί λιγότερο από 8MB μνήμης ανεξαρτήτως της εντολής ή αρχείου ήχου που δίνεται.

- **Περιορισμοί Χρόνου**. Το πρόγραμμά σας πρέπει να μπορεί να διαχειρίζεται αρχεία των 10MB σε λιγότερο από 1 δευτερόλεπτο.

- **Περιορισμοί Εισόδου**. Το πρόγραμμά σας πρέπει να μπορεί να διαχειρίζεται ακολουθίες ήχου wav *οποιουδήποτε μεγέθους*. Το κάθε byte εισόδου θα μπορεί να διαβαστεί από την πρότυπη είσοδο *μόνο μία φορά* (δεν θα είναι εφικτή η χρήση fseek) χρησιμοποιώντας την γνωστή μας `getchar`.

- Απαγορεύεται ρητά η χρήση δομών (structs) σε αυτήν την εργασία.

Η Υποεντολή info (40% της βαθμολογίας)

Όταν το πρόγραμμά σας κληθεί με την υποεντολή info ως πρώτο όρισμα, δηλαδή ως `./soundwave info` θα πρέπει να ελέγχει αν τα δεδομένα που διαβάστηκαν ακολουθούν το πρότυπο wav που περιγράφηκε παραπάνω και να εκτυπώνει τις σχετικές πληροφορίες του ήχου (χωρίς τα δεδομένα αυτά καθεαυτά). Παράδειγματα εκτέλεσης ακολουθούν:

```
$ ./soundwave info < good.wav
size of file: 133
size of format chunk: 16
WAVE type format: 1
mono/stereo: 1
sample rate: 44100
bytes/sec: 88200
block alignment: 2
bits/sample: 16
size of data chunk: 88
$ ./soundwave info < 1MB.wav
size of file: 1073210
size of format chunk: 16
WAVE type format: 1
mono/stereo: 2
sample rate: 8000
bytes/sec: 32000
block alignment: 4
bits/sample: 16
size of data chunk: 1072948
$ ./soundwave info < 2MB.wav
size of file: 2146158
size of format chunk: 16
WAVE type format: 1
mono/stereo: 2
sample rate: 16000
bytes/sec: 64000
block alignment: 4
bits/sample: 16
size of data chunk: 2145896
$ ./soundwave info < 5MB.wav
size of file: 5226758
size of format chunk: 16
WAVE type format: 1
mono/stereo: 2
sample rate: 44100
bytes/sec: 176400
block alignment: 4
bits/sample: 16
size of data chunk: 5226496
$ ./soundwave info < 10MB.wav
size of file: 10406730
```

```
size of format chunk: 16
WAVE type format: 1
mono/stereo: 2
sample rate: 44100
bytes/sec: 176400
block alignment: 4
bits/sample: 16
size of data chunk: 10406468
```

Όλα τα αρχεία που φαίνονται στα παραδείγματα της εκφώνησης της εργασίας μπορείτε να τα κατεβάσετε από το https://github.com/progintro/wav.

Οι εκτυπώσεις του προγράμματός σας θα γίνονται κανονικά στην πρότυπη έξοδο, **εκτός** αν παρουσιαστεί κάποιο σφάλμα (δείτε παρακάτω) οπότε και θα πρέπει να τυπώσετε στο πρότυπο ρεύμα εξόδου σφάλματος (stderr / standard error). Για να τυπώσετε στο stderr, μπορείτε να χρησιμοποιήσετε την συνάρτηση fprintf με πρώτο όρισμα το stderr εξής:

`fprintf(stderr,` <ίδια ακριβώς ορίσματα με αυτά της `printf`>);

Το πρόγραμμά σας πρέπει να είναι σε θέση να καταλαβαίνει λάθη που υπάρχουν στα δεδομένα ήχου που διαβάστηκαν και να σταματά στο πρώτο λάθος που βρίσκει εκτυπώνοντας το κατάλληλο διαγνωστικό μήνυμα στο stderr. Παραδείγματα εκτέλεσης σε προβληματικά δεδομένα εισόδου είναι τα ακόλουθα:

```
$ ./soundwave info < bad_riff.wav
Error! "RIFF" not found
$ echo $?
1
$ ./soundwave info < bad_riff.wav 2> error.txt
$ cat error.txt
Error! "RIFF" not found
$ ./soundwave info < bad_wave.wav
size of file: 133
Error! "WAVE" not found
$ ./soundwave info < bad_fmt.wav
size of file: 133
Error! "fmt " not found
$ ./soundwave info < bad_sfc.wav
size of file: 133
size of format chunk: 18
Error! size of format chunk should be 16
$ ./soundwave info < bad_wtf.wav
size of file: 133
```

```
size of format chunk: 16
WAVE type format: 2
Error! WAVE type format should be 1
$ ./soundwave info < bad_ms.wav
size of file: 133
size of format chunk: 16
WAVE type format: 1
mono/stereo: 3
Error! mono/stereo should be 1 or 2
$ ./soundwave info < bad_bys.wav
size of file: 133
size of format chunk: 16
WAVE type format: 1
mono/stereo: 1
sample rate: 44100
bytes/sec: 88202
block alignment: 2
Error! bytes/second should be sample rate x block alignment
$ ./soundwave info < bad_bis.wav
size of file: 133
size of format chunk: 16
WAVE type format: 1
mono/stereo: 1
sample rate: 44100
bytes/sec: 88200
block alignment: 2
bits/sample: 32
Error! bits/sample should be 8 or 16
$ ./soundwave info < bad_ba.wav
size of file: 133
size of format chunk: 16
WAVE type format: 1
mono/stereo: 2
sample rate: 44100
bytes/sec: 88200
block alignment: 2
bits/sample: 16
Error! block alignment should be bits per sample / 8 x mono/stereo
$ ./soundwave info < bad_data.wav
size of file: 133
size of format chunk: 16
WAVE type format: 1
```

```
mono/stereo: 1
sample rate: 44100
bytes/sec: 88200
block alignment: 2
bits/sample: 16
Error! "data" not found
$ ./soundwave info < bad_insfd.wav
size of file: 133
size of format chunk: 16
WAVE type format: 1
mono/stereo: 1
sample rate: 44100
bytes/sec: 88200
block alignment: 2
bits/sample: 16
size of data chunk: 88
Error! insufficient data
$ ./soundwave info < bad_sf.wav
size of file: 128
size of format chunk: 16
WAVE type format: 1
mono/stereo: 1
sample rate: 44100
bytes/sec: 88200
block alignment: 2
bits/sample: 16
size of data chunk: 88
Error! bad file size (found data past the expected end of file)
```

Στο τελευταίο παράδειγμα εκτέλεσης, παρατηρούμε ότι το `bad_sf.wav` αρχείο ισχυρίζεται ότι έχει μέγεθος 128 bytes (σύνολο 136 με τα αρχικά 8 bytes της κεφαλίδας), όμως αν ελέγξουμε τα συνολικά bytes του αρχείου:

```
$ wc -c ./bad_sf.wav
141 ./bad_sf.wav
```

παρατηρούμε ότι έχει και άλλα bytes στο τέλος τα οποία δεν συμπεριλαμβάνονται στο SizeOfFile και επομένως δεν περνάει τον έλεγχο ορθότητας (validation). Αντίστοιχα, μπορείτε να φτιάξετε και εσείς τα δικά σας παραδείγματα με λανθασμένα wav προκειμένου να ελέγξετε ότι το πρόγραμμά σας τα αναγνωρίζει επιτυχώς.

Η υποεντολή rate (40% της βαθμολογίας)

Τώρα που το πρόγραμμά μας μπορεί να αναγνωρίζει αρχεία wav, ήρθε η ώρα να προσθέσουμε την πρώτη sound-editing υποεντολή μας, την υποεντολή rate, της

οποίας η σύνταξη είναι `./soundwave rate fp_rate`, όπου `fp_rate` είναι ένας αριθμός κινητής υποδιαστολής (double) που αντιστοιχεί στην επιτάχυνση που επιθυμούμε στον ρυθμό αναπαραγωγής. Η υποεντολή rate, εκτός από τον έλεγχο ορθότητας των δεδομένων εισόδου θέλουμε να μεταφέρει στην έξοδο, με τη βοήθεια της συνάρτησης putchar, τα δεδομένα του ήχου, έχοντας αλλάξει την ταχύτητα αναπαραγωγής του, χωρίς να αλλοιώσει τα δείγματα που αποτελούν τον ψηφιοποιημένο ήχο.

Αν το υλοποιήσουμε σωστά, καλώντας το πρόγραμμά μας ως `./soundwave rate 0.5 < input.wav > output.wav` περιμένουμε το output.wav αρχείο να αναπαράγεται με υποδιπλάσιο ρυθμό, δηλαδή στον διπλάσιο χρόνο από τον αρχικό. Σκεφτείτε ποιες παράμετροι των δεδομένων εισόδου θα πρέπει να αλλάξουν και πώς, ώστε να επιτευχθεί το ζητούμενο. Σε κάθε περίπτωση, η έξοδος του προγράμματος πρέπει να είναι μία νόμιμη ακολουθία από δεδομένα wav. Ακόμα και αν υπάρχουν στην είσοδο επιπλέον bytes μετά το τμήμα των δεδομένων του (OtherData), θα πρέπει και αυτά να μεταφερθούν στην έξοδο, προφανώς αυτούσια. Παραδείγματα εκτέλεσης ακολουθούν:

```
$ ./soundwave info < LaughEvil.wav
size of file: 329266
size of format chunk: 16
WAVE type format: 1
mono/stereo: 1
sample rate: 44100
bytes/sec: 88200
block alignment: 2
bits/sample: 16
size of data chunk: 329230
$ ./soundwave rate 0.5 < LaughEvil.wav > LaughEvil2.wav
$ ./soundwave info < LaughEvil2.wav
size of file: 329266
size of format chunk: 16
WAVE type format: 1
mono/stereo: 1
sample rate: 22050
bytes/sec: 44100
block alignment: 2
bits/sample: 16
size of data chunk: 329230
$ ./soundwave rate 0.5 < ThePinkPantherTheme.wav > ThePinkPantherTheme2.wav
$ ./soundwave rate 0.5 < la.wav > la2.wav
$ ./soundwave rate 0.5 < 8bitstereo.wav > 8bitstereo2.wav
```

Στα παραπάνω παραδείγματα, η έξοδος του προγράμματος ανακατευθύνεται σε ένα νέο αρχείο ήχου. Αν θέλετε να αναπαράξετε αρχεία ήχου, ο εύκολος τρόπος σε

κάθε υπολογιστή και λειτουργικό σύστημα είναι να κάνετε διπλό κλικ στο αρχείο. (Σχεδόν) πάντοτε υπάρχει για τα αρχεία ήχου μία εφαρμογή που είναι συνυφασμένη με αυτά, ώστε με διπλό κλικ στα αρχεία να γίνεται η αναπαραγωγή που θέλουμε.

Σε κάποια συστήματα, η αναπαραγωγή αρχείων ήχου μπορεί να γίνει και με κατάλληλη εντολή στο τερματικό, για παράδειγμα την play για Linux συστήματα, ή την afplay για MacOS. Μάλιστα, σε Linux μπορεί να αποφευχθεί η δημιουργία αρχείου με το τελικό αποτέλεσμα, αν απλώς κάνουμε σωλήνωση. Για παράδειγμα, `./soundwave rate la.wav | play -`. Ή, ακόμα πιο απλά, για ένα απομακρυσμένο αρχείο ήχου:

```
curl -s https://raw.githubusercontent.com/progintro/wav/refs/heads/main/la.wav |
  ./soundwave rate 0.5 | play -
```

Αντίστοιχα, μπορείτε να χρησιμοποιήσετε την παράμετρο της υποεντολής rate προκειμένου να *διπλασιάσετε* την ταχύτητα αναπαραγωγής του. Διπλασιάζοντας την ταχύτητα αναπαραγωγής (2x speedup), ο χρόνος αναπαραγωγής θα υποδιπλασιασθεί, αφού τα δείγματα που αποτελούν τον ψηφιοποιημένο ήχο θα πρέπει και πάλι να μεταφερθούν αναλλοίωτα. Παραδείγματα εκτέλεσης ακολουθούν:

```
$ ./soundwave info < LaughSarc.wav
size of file: 1541540
size of format chunk: 16
WAVE type format: 1
mono/stereo: 2
sample rate: 48000
bytes/sec: 192000
block alignment: 4
bits/sample: 16
size of data chunk: 1541504
$ ./soundwave rate 2 < LaughSarc.wav > LaughSarc3.wav
$ ./soundwave info < LaughSarc3.wav
size of file: 1541540
size of format chunk: 16
WAVE type format: 1
mono/stereo: 2
sample rate: 96000
bytes/sec: 384000
block alignment: 4
bits/sample: 16
size of data chunk: 1541504
$ ./soundwave rate 2.0 < ThePinkPantherTheme.wav > ThePinkPantherTheme3.wav
$ ./soundwave rate 2.0 < narc.wav > narc3.wav
$ ./soundwave rate 2.0 < narc.wav | ./soundwave info
size of file: 4956196
```

```
size of format chunk: 16
WAVE type format: 1
mono/stereo: 2
sample rate: 88200
bytes/sec: 352800
block alignment: 4
bits/sample: 16
size of data chunk: 4956160
$ ./soundwave rate 2.0 < 8bitmono.wav | tee 8bitmono3.wav | ./soundwave info
size of file: 44136
size of format chunk: 16
WAVE type format: 1
mono/stereo: 1
sample rate: 88200
bytes/sec: 88200
block alignment: 1
bits/sample: 8
size of data chunk: 44100
```

Η υποεντολή channel (20% της βαθμολογίας)

Η υποεντολή channel επιτρέπει στον χρήστη να διαλέξει να αναπαράξει δεδομένα μόνο από το αριστερό ή δεξί κανάλι ήχου στην περίπτωση που ο ήχος της εισόδου είναι στερεοφωνικός. Η σύνταξη της εντολής είναι ως εξής: `./soundwave channel left` για να κρατήσει μόνο τα δεδομένα από το αριστερό κανάλι και `./soundwave channel right` για να κρατήσει μόνο τα δεδομένα από το δεξί κανάλι. Όπως προηγουμένως, το πρόγραμμά σας πρέπει να πραγματοποιεί έλεγχο ορθότητας των δεδομένων εισόδου και μεταφέρει στην έξοδο, με τη βοήθεια της συνάρτησης putchar, τα δεδομένα μόνο του καναλιού του ήχου που ζήτησε ο χρήστης. Αν ο ήχος στην είσοδο είναι μονοφωνικός, να μην προκύπτει σφάλμα, αλλά απλώς να μεταφέρεται στην έξοδο το μοναδικό κανάλι της εισόδου. Σκεφτείτε ποιες παράμετροι των δεδομένων εισόδου θα πρέπει να αλλάξουν και πώς και ποια από τα bytes του τμήματος δεδομένων της εισόδου θα πρέπει να μεταφερθούν στην έξοδο, ώστε να επιτευχθεί το ζητούμενο. Σε κάθε περίπτωση, η έξοδος του προγράμματος πρέπει να είναι μία νόμιμη ακολουθία από δεδομένα που συνιστούν έναν ήχο με βάση το πρότυπο που έχει περιγραφεί. Ακόμα και αν υπάρχουν στην είσοδο επιπλέον bytes μετά το τμήμα των δεδομένων του (OtherData), θα πρέπει και αυτά να μεταφερθούν στην έξοδο, προφανώς αυτούσια.

Παραδείγματα εκτέλεσης ακολουθούν:

```
$ ./soundwave info < 8bitstereo.wav
size of file: 176436
size of format chunk: 16
```

```
WAVE type format: 1
mono/stereo: 2
sample rate: 44100
bytes/sec: 88200
block alignment: 2
bits/sample: 8
size of data chunk: 176400
$ ./soundwave channel left < 8bitstereo.wav | tee 8bitstereo4.wav | ./soundwave info
size of file: 88236
size of format chunk: 16
WAVE type format: 1
mono/stereo: 1
sample rate: 44100
bytes/sec: 44100
block alignment: 1
bits/sample: 8
size of data chunk: 88200
$ ./soundwave channel left < CaliforniaDreamin.wav > CaliforniaDreamin4.wav
$ ./soundwave channel right < 8bitstereo.wav > 8bitstereo5.wav
$ ./soundwave channel right < CaliforniaDreamin.wav > CaliforniaDreamin5.wav
```

Η υποεντολή volume (Bonus 20% της βαθμολογίας)

Η υποεντολή volume δέχεται έναν πολλαπλασιαστή για την ένταση του ήχου με την σύνταξη `./soundwave volume fp_multiplier` και παράγει ένα καινούριο αρχείο ήχου όπου η ένταση κάθε δείγματος έχει πολλαπλασιαστεί με τον αριθμό κινητής υποδιαστολής `fp_volume`. Για παράδειγμα, αν το πρόγραμμά σας κληθεί ως εξής: `./soundwave volume 0.125` περιμένουμε ότι το αρχείο ήχου εξόδου θα έχει το 1/8 της έντασης του αρχικού αρχείου. Παραδείγματα εκτέλεσης ακολουθούν:

```
$ ./soundwave volume 0.125 < 8bitmono.wav > 8bitmono6.wav
$ ./soundwave volume 0.125 < 8bitstereo.wav > 8bitstereo6.wav
$ ./soundwave volume 0.125 < la.wav | tee la6.wav | ./soundwave info
size of file: 88236
size of format chunk: 16
WAVE type format: 1
mono/stereo: 1
sample rate: 44100
bytes/sec: 88200
block alignment: 2
bits/sample: 16
size of data chunk: 88200
```

Η υποεντολή generate (Bonus 30% της βαθμολογίας)

Οι παραπάνω εντολές επιτρέπουν στο πρόγραμμά μας να επεξεργάζεται υπάρχοντα αρχεία ήχου. Τι κάνουμε όμως αν εμείς θέλουμε να παράξουμε τους δικούς μας ήχους; Σε αυτό ακριβώς το πρόβλημα έρχεται να δώσει λύση η υποεντολή generate. Η σύνταξη της εντολής generate είναι `./soundwave generate` και παράγει στην έξοδο δεδομένα τύπου wav, με βάση τον εξής μαθηματικό τύπο:[1]

$$f(t) = amp \cdot \sin(2\pi f_c t - m_i \cdot \sin(2\pi f_m t))$$

Για το σκοπό αυτό, ορίστε μία συνάρτηση C με πρωτότυπο το

```
void mysound(int dur, int sr, double fm, double fc, double mi, double amp);
```

η οποία να καλείται με κατάλληλες τιμές στα ορίσματά της από τη main συνάρτηση, ώστε να παράγει τον κατάλληλο ήχο με βάση τον μαθηματικό τύπο που δόθηκε. Η υποεντολή, πέρα από τον βασικό τρόπο με τον οποίο καλείται (χωρίς άλλα ορίσματα) πρέπει να μπορεί να υποστηρίζει τα ακόλουθα προαιρετικά ορίσματα για τον καθορισμό των παραμέτρων:

- `--dur int_duration`: Το όρισμα dur είναι η διάρκεια του ήχου σε δευτερόλεπτα (ακέραιος). Default: 3.
- `--sr int_sr`: Το sr είναι το *SampleRate* για την ψηφιοποίηση του ήχου (ακέραιος). Default: 44100.
- `--fm fp_fm`: Το frequency modulation που χρησιμοποιείται στο σήμα f(t) (κινητής υποδιαστολής). Default: 2.0.
- `--fc fp_fc`: Το carrier frequency που χρησιμοποιείται στο σήμα f(t) (κινητής υποδιαστολής). Default: 1500.0.
- `--mi fp_mi`: Το modulation index που χρησιμοποιείται στο σήμα f(t) (κινητής υποδιαστολής). Default: 100.0.
- `--amp fp_amp`: Το amplitude (δηλαδή η ένταση) που χρησιμοποιείται στο σήμα f(t) (κινητής υποδιαστολής). Default: 30000.0.

Οι παραπάνω παράμετροι πρέπει να τηρούνται πλήρως κατά την παραγωγή του ηχητικού σήματος. Για το π μπορείτε να χρησιμοποιήσετε τη συμβολική σταθερά M_PI, που είναι ορισμένη στο math.h. Ο ήχος που θα παραχθεί να είναι μονοφωνικός (*MonoStereo* = 1) και να αναπαρίσταται με 2 bytes ανά δείγμα (*BitsPerSample* = 16).

Καλέστε το πρόγραμμά σας με τις default παραμέτρους. Τι ήχος παράγετα; Ενδεικτική εκτέλεση:

[1] Το π είναι το γνωστό 3.14... και με το sin συμβολίζεται η συνάρτηση του ημιτόνου.

```
$ ./soundwave generate > mysound.wav
$ ./soundwave info < mysound.wav
size of file: 264636
size of format chunk: 16
WAVE type format: 1
mono/stereo: 1
sample rate: 44100
bytes/sec: 88200
block alignment: 2
bits/sample: 16
size of data chunk: 264600
./soundwave generate --dur 5 --fm 1.2 --fc 300 --mi 100 --amp 15000 > space.wav
```

Διαγωνισμός Δημιουργικότητας και Τελική Υποβολή

Οι παραπάνω υποεντολές μας προσέφεραν την δυνατότητα να διαχειριστούμε δυαδικά αρχεία ήχου και είδαμε παραδείγματα χρήσης τους. Από εκεί και πέρα, σας συνιστούμε να πειραματιστείτε και με άλλες δυνατότητες—για παράδειγμα την προσθήκη ορισμάτων --help ώστε να μπορεί να βρούμε μια γρήγορη περιγραφή των διαφόρων δυνατοτήτων του προγράμματός σας.

Για να λάβετε συμμετοχή στον διαγωνισμό δημιουργικότητας στο τέλος του εξαμήνου, μπορείτε να υλοποιήσετε μια δική σας υποεντολή `./soundwave dj` (με όποια ορίσματα θέλετε) η οποία θα είναι σε θέση να παράξει ενδιαφέροντες και δημιουργικούς ήχους! Αν κάποια υποβολή μας εκπλήξει, θα λάβει επιπλέον bonus μέχρι 50% της βαθμολογίας.

Τέλος, πριν την υποβολή της εργασίας, μην ξεχάσετε το soundwave/README.md, μέσα στο οποίο πρέπει να συμπεριλάβετε τις όποιες παρατηρήσεις σας κατά την διεκπεραίωση της άσκησης. Ο κώδικας απαιτείται να είναι καλά τεκμηριωμένος με σχόλια καθώς αυτό θα είναι μέρος της βαθμολόγησης.

Καλή Επιτυχία!

Αναφορές

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

[2] DIT. Πρόσκληση για Εργασία 1 . https://classroom.github.com/a/Ej4U3Ftp.

[3] fileformat. Waveform File Audio. https://docs.fileformat.com/audio/wav/.

[4] Wikipedia. https://en.wikipedia.org/wiki/Endianness.

[5] Wikipedia. 44,100 Hz frequency. https://en.wikipedia.org/wiki/44,100_Hz.

[6] Wikipedia. Compact Disc (CD). https://en.wikipedia.org/wiki/Compact_disc.

[7] Wikipedia. The WAV File Format. https://en.wikipedia.org/wiki/WAV.

[8] YouTube. How much a shadow weighs and what is the speed of "push"? (αν ξεκινήσατε την άσκηση μια μέρα - ή ώρα - πριν την υποβολή συνιστούμε να μην πατήσετε το λινκ). https://www.youtube.com/watch?v=Do1lm9IevYE.