סוגי מידע

מידע → { מספרי, תווים, לוגי }

כדי לייצג את המספרה 2: 0 1 0 → 1, 2, 4

3: 0 1 1 → 1×1 + 2×1 + 4×0 = 3

לייצג מספר ב- 1 בית ב- Binary

1: 001 | 4: 100 | 7: 111

2: 010 | 5: 101 | 8: ?

3: 011 | 6: 110 | 9: ?

bit = binary digit ← ביט אחד יכול לייצג 2 מצבים 0 או 1

byte ← 8 bits אין מחשב שמעבד ביט אחד

הספרות של bit: 2^0 2^1 2^2 2^3 2^4 2^5 ...

10: 1010 | 9: 1001 | 8: 1000

איך: מעגלים

במחשב יש טרנזיסטורים (transistor) ← רכיב (במחשב יש מיליון או מיליארד כולה) שמסוגלים לייצג מידע ולאחסן ערכים (Store values). כל מחשב בנוי לייצג את המספרה 2, כלומר יהיו 2 טרנזיסטורים שלה את המספרה 2 ⇐ טרנזיסטור 1 כלומר לכל טרנזיסטור במחשב שולח קצת חשמל (on=מס) ומוסר חשמל (off=0)

איך במחשב מייצגים תווים? הכללה: מחרוזת? כדי לייצג אותיות בשפה

ASCII

A B C D E F G ...

65 66 67 68 69 70 71

כמובן צריך שיהיה לייצג גם את השפות שלנו כמו עברית ואנגלית כמו à, é, ! . א.

לכן יש בעיה בהרחבה של השיטה הקיימת כמו ובמו Unicode

ה- נפוצים בקידוד (בתים) כמו תמונות אבל מחרוזות/מס' ... 0 1111 = מ-128 = אות / מספר = (אחרת שמה)

מה לגבי תמונות? הרי הסולמי שם אם כדי לשלוח אותו אני משתמש ב-...0000111 - איך יפרסו אותו?

באמצעות RGB ופיקסלים. כל פיקסל של צבעים של צבעים 33 73 72 נקי Hi אבל בקונקטים של פרוטוקולים

הוא נקי פיקסל אחד. כל פיקסל מורכב ממספר של צבעים, ירוק וכחול.

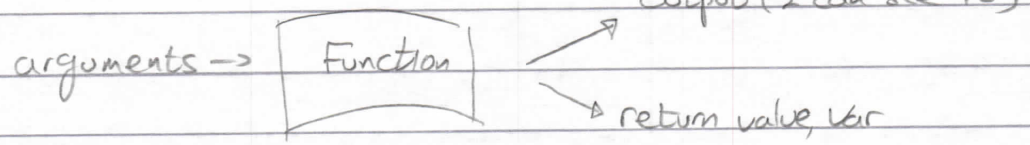
וכך: תמונה + תמונה + תמונה + תמונה + תמונה מורכבת מאוסף וולכים את שלמתן תוספים כמעט.

1. C

Correctness Design Style

Correctness: does it work? does it do what it suppose to do?
Design: How well is your code written? efficent code, doesnt repeat itself
Style: קלות. good visibility and more readable

Functions: verbs, actions. can take inputs => arguments, parameters



header file: import -> methods

tools for excercies:
help 50 style50
commands: Terminal window
ls - list the contact in the current folder x = exec file
rm - remove mv - rename (move) -> mv hello.c goodbye.c and move...
mkdir - create folder mv - move -> move hello.c lecture/
cd - change directory move one folder up: mv xxx.c ..
rmdir - remove directory (folder) foldername/
../ -> parent folder ./ -> current folder
make - compile (not really, we wild see it later)

Types

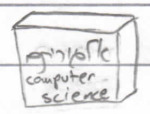
int 32 bits can represent 4,000,000,000 -> -2,000,000,000 - 2,000,000,000
print -> %c = char %f = float %i = int %li = long %s = String
long 64 bits

int x 1 / int y 2 = 0

float z = x/y
z = 0.000 and not z = 0.5

float z = x/y x/y != 0.5 -> x/y == 0 and that's why when I put it inside a float ref -> z == 0.000

מוסיקה: שם כזה ניתנה בשנה שבה יש אלגוריתמים ל- Duration ו- Volume. הכל מתקבל ל-1: 0



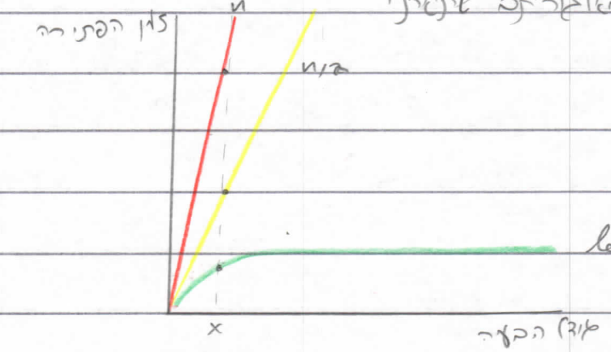
כל השיר מוצלח שלפחות קלידים יכולים להכניס בתוך input. את לגבי ה- output

כותרת

אלגוריתם: הדמיה בעזרת אור בעזרת אלגוריתם לחיבור מספרים. אני יכול להביא ספר מלפונים - אני רוצה למקם אלגוריתם לחיבור של John Levi בספר מלפונים. אני יכול להביא ספר מלפונים אחר שבו יש שמות לפי A ומספרים לפי הגיל. אם יש אלפיים שמות - כמה הוא יחזיר? אולי - לא. אני נניח שאני מחפש אדם שנקרא ב- A ואני מתקבל $2^0=1$ פעמים ואם לא מוצאים אותו אני 2^1 פעמים ואז 2^2 פעמים 2^3 וכו'. הוא הוא יחזיר לי: כמה. הוא הוא מוכן לעבוד. בדיוק הצעד אני יכול לבטא את John Levi. מפורד לפחות אחרי. השורה מה הקבוצה של 2^x אבל במילים ואחרי את ... על השורה אפשר אני יכול לקחת את המשפט לתכור בלי אחרים אך לחיבור הוא. בספר הטלפונים יש 4,500 דפים. השורה הראשונה יתן לי בערך 5000 דפים אך לחיבור הוא, כלומר הוא יתן לי פחות ממה שהוא צריך. אני יכול, כי- 5000 דפים מלאך לחיבור את הוא. כיצד? אם בספר יש 4,500 דפים/לפחות - אני אחלק ב-2 את שיעור ל-1. אני אסתכל את הספר באמצע. הוא John Levi נמצא בחצי הראשון/השני של הספר. אחלק את החלק הזה ואני ניעזר עם 2015 עליו שוב אסתכל באמצע, וכך: $1 \leftarrow 2 \leftarrow 4 \leftarrow 8 \leftarrow 16 \leftarrow 32 \leftarrow 64 \leftarrow 128 \leftarrow 256 \leftarrow 512 \leftarrow 4,500$

בתפיסת אלגוריתם אם לקחת את הנקודה שלך ואלך היית עולה למטה בבניין בלי ולחשוב את מה לעשות שמאל / ואז שוב ימין להמין.

מכאן יש שאלת האלגוריתם שתיארנו



אחרי רוצים שלפיו בתה בחוברת נתון x-מספר הבתים ושאלה ושאלות בקלות באלגוריתם.
בנוסף בכאן פתיחה נתון, אחרי יכולים לפתור בעזרת אחרת יותר גדולות את שאלות באלגוריתם על ידי. בשלוש הינו אלגוריתם אחרת אך נקח היותר, שבו בעזרת אחרת נפתרות בכאן מהה לפעמים פשוטים.

דוגמה: אני לוקח ספר טלפונים אחר שבו יש בערך 4,500 דפים, כמה יוסף לי לקח אחד בשביל האלגוריתם ויספור לי. כמה אחרות אך לחלק בדפים

Loops

while forever => while (true)

Scope

the scope of a var is inside the { }

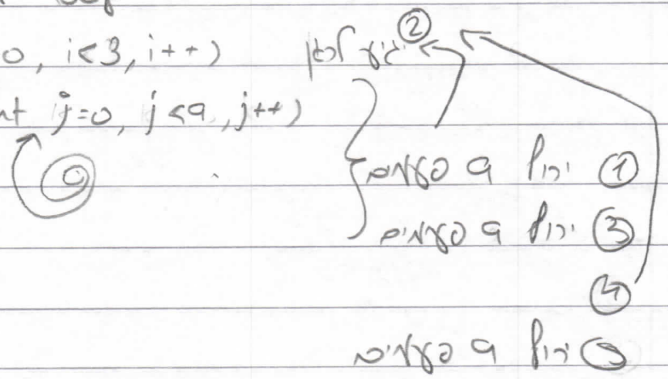
if it's inside the main{ } it's accesible as long as the main run

if it's inside of a function( ){var} or loop( ){var} it's accesible only when the function or the loop are active on the stack (on the top)

a soultion for this - if it's a problem is to declare the var outside the { }

Nested for loop

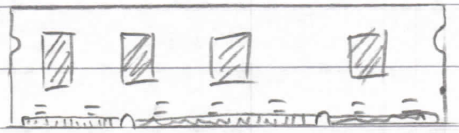
```
for(int i=0, i<3, i++)
    for(int j=0, j<9, j++)
```



run 9 x 3 → i · j => times

Limitation

RAM



כיכון אינסוף

יש ∞

to store int I use 32 bits, float also.

המיכון הוא מוגבל. יש לו כמות קבועה של ביטים ← אין יכול להתקיים מיכון שאינו יש לו אינסוף. כלומר אחרי יש יכול להיות (בד"כ) המוגבה ביותר שלו) אין אפשרי לשמור

1/10 = 0.10 BUT → 0.100000056160007800

code

זה היינו כאילו קוד pseudocode לחפש John Smith

```
1  pick up phone Book
2  open to the middle of the phone Book
3  Look at the page
4  if Smith is on page
5     call Mike
6  Else if Smith is earlier on the Book
7     open to middle of left half of Book
8     Go back to line 3
9  Else if Smith is later in Book
10    open to middle of right half of Book
11    Go back to line 3
12 Else                    → אם הוא לא נמצא בספר
13    quit
```

מקרא:
functions
conditions
Boolean expressions
loops

function: כשרוצים לפצל, לחלק למשימה יותר קטנה

conditions: נקי בגדול בקוד. אלה כולן קטעים שבהם יש אם או אם לא < אבל כל התנאים שלה בקוד לשלוח אל בלוק שלה

exp Boolean: משהו שמחזיר בקוד לשלוח אל בלוק אחר שהוא מחזיר בשביל בקוד לבצע. שאלות שיש כן/לא, True/false, 0/1

loop: קשה לי את אלו הדברים שיש לולאה

scratch

הבעיה שהיה מרובעים בצורה וצבועה עם אריחים של blocks שנראים כמו חלקי פאזל

כשרוצים שורת/שורות קוד ורוצים שהן תתבצענה עוד פעמון יש אפשרות ל- copy/paste כי אז הקוד יתארכן

יש אפשרות ל- repeat

How to Programme

1:

המחשב עבדן טיפה בקלבית שלה ביחוס ל 1 ו 0, כשאנחנו אומרים למחשב "פקודות" או "הוראות". המחשב מבין בעיה מסוימת/מטלות הוראות כדי לבצע, לכן לפעמים צריך לפרק לו הכל לפי הוראות יחסי, ומשהו בסגנון כשאנחנו מדברים על שפת ה-High level ניגשים וכלי לתכנת קוד שיהיה מובן לו אלגוריתם מסוים שיש במחשב שלו יחסי. בשביל שמחשב יבין לך, המחשב, באמצעות compile או interpeter תורגם את הקוד לשפה של 1 ו 0.

interpet: לוקח שורה מהקוד ומריץ אותה איך

compiler: לוקח את כל הפעולה בפעם אחת ומסיים פקודות המחשב מהר יותר לשפת לפני את הקוד

ביתון מיוחס כשפה שהוא interpet אבל לוגלה עשוי ב-compiler למרבית מהי interpet

2:

The element of computing systems: ספר שמסביר על מנת להסביר איך התוכנה עובדת במחשב

שלמעשה הוא מהי קומפיוטר מה מרכיבים אותו ואיך הם עובדים.

statement: שילוב של operations או פקודות I/O (Input/Output)

תיבעות צה מרכיבים של קומפיוטר: הלוגיקת הקומפיוטר, הזכרון שמרים בעיקר, שליח שמרים מעיך הקומפיוטר ובין קומפיוטר לקומפיוטר.

טקס: כל הפעולות של המחשב מעבדות שם

זיכרון: לוקח המח"ה של ה cache. הוא מכיל מידע שמחשב שתמלוקות לקבוצות בקלות בלוא הן קרוצות ל-טקס כל שצה יותר קרוב ל- טקס, כך ל- טקס יהיה יותר קל ויותר מהיר לגשת לזיכרון.

כל שלוי מתקדם מה - טקס כמה לוקח פחות צמן לגשת לזיכרון, לכן יש ל יותר מהני ובשביל כלי הוא מכיל מעבד יותר צמן, לשלי דיסק קשה (HDD, SSD) לצמת RAM (cache) לצמת cache (מה שיכה registers) לצמת CPU

HDD ← RAM ← cache memory ← registers ← CPU

* מעבד ה - טקס יש יתרת זיכרון שנקרא registers מבחינת זליות קלה

* ה - cache נועד קרוב לליבת ה-טקס והוא בשלוי מעבד גשוב מאוד כמו ה - טקס

* שם מתקדים עוד מה - טקס מבדילים את ה-primary memory. מה זיכרון "שמוה" פה שומרים את כל הנתונים ש"בזים" כמו תיבעת שפונקות, אך הפעלה. שם המחשב נמצה בפתאומיות הנתונים בזיכרון לבדו.

* שם עוד מתקדים מה - טקס מיקה ל- secondary memory = זכרון: דיסק קשה / SSD. בזה מאחסנים אף לשמור צמן כמו קבצים. התיבעת מהירות איזה ניץ לוקח מהזיכרון הצמן: (read) ולהמיך שמו (write) בקבצים הנמצאים בזיכרון לעיוות ארוך.

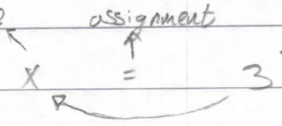
* tertiary memory: זכרון ברוך. נתונים שנמצאים רי אך המחשב זהו כמשל, אך שמת cloud. אומרו שכל שהזיכרון מתקדם המחשב כך יש יותר מהני. כמו זיכרון הקבלה לנשמ דיסק קשה (הכי "מיתה" כמו 15TB).

זיכרון צמן הוא כמו סט של קומפיוטים, כל קומפיוט יכולה להכיל מידע. שם אני רוצה מילה לאחת מהקומפיוטים הללו אני צריך לשים שלה, לכל קומפיוט יש שם. לקומפיוט הצמת אלהן מתייחסים בשם Variable

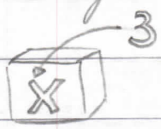
var

מה בעצם var בשפה, אפשר להקצות לו פעם אחת. כל var יכול לשמור מידע

שם הקופסא assignment מידע



assignment



x=3 x=3.4 x="make me a pizza"

קבוצה של var בשלמו: אלינת, אספרים וקו יתחתון. אסור שיתחיל במספרים ואסור שיהיה בהם רווח שקובה

הפעלה קובע מיידית הקצאה (שים את y ב-[x]) x=y (int, float, String)

operations

הפעולות הכי פשוטות שהמחשב יכול לעשות זה אריתמטיקה +, -, /, * => כמה בעיו ב- שפה

אפשר להשתמש במשתנה הללו כמו בכל var אחרים. a=2+5 -> b=a-4 -> c=a+b

concatenation: פעולה שמאפשרת לו string = חיבור 2 מחרוזת תוך שימוש ב- (+) s1="Hellowo"

s2="ו R wo How?"

concatenation: s3=s1+s2 => s3="Hellowo ו R wo?"

ב- String אין פעולות של חילוק, חיסור או הכפלה

בפעולות בין מספרים שונים של var זה קורס שווה. כי אפשר לחבר String ו- int אבל פעולה לא תעבוד. בפייתון יש יוצא דופן אפשר להכפיל String במספר int כפי לחצור לפי ה- String x פעמים

input/output

SQL

#sqlite

sql: language that is used to querel the data base. Do 4 things

Create — CREATE, INSERT | Read — SELECT | Uplate — UPDATE | Delete — DELETE

SELECT

SELECT 1 or more colums FROM table.name

SELECT *: all of the colums

SELECT DISTINCT (colum.n) From table.n — מסיר את ה-duplicates

WHERE → condition = true / false

SELECT title FROM table.n WHERE title=" את שם ה- " → ex "the office"

SELECT -"- -"- title LIKE "%office%"

כל מה שאחרי ה-% יכול להיות שווה ל-0 או יותר chs כלשהם

SELECT DISTINCT(UPPER(title) FROM Shows ORDER BY UPPER(title);

אומר לו את כל ה-value בתוך ה-colum ← title, שים הכל ב-value ואחרי זה תעשה לזה uppercase.

תמיין הלבלה שנקראת shows ותמיין את זה לפי סדר ABC (יתבצע רק ב-upper)

let's try to count how many occurance of the same value we have in the DB

SELECT UPPER(title), COUNT(title) FROM shows GROUP BY UPPER(title);

אם אתה רוצה שנדע מה זה עושה לפי הקודם (פופולרים) לפי אופן ↓

ORDER BY COUNT(title)

DESC ↓+

ASC — סדר מהקטן לגדול, זה ברירת המחדל

LIMIT 10 — זה יגביל כך את ה-10 התוצאות הראשונים

TRIM(title) — קיצץ רווחים מיותרים לפני ואחרי הלבלה

הוספת מידע

INSERT INTO table(column, ...) VALUES (value, ...) — הוספת שורה

UPDATE shows SET genres = " " WHERE title = "..." — הוספה לפי ה-colum

מחיקת מידע

DELETE FROM table.n WHERE (אחד) title = "_ _ _"

Data Type

BLOB → Binary Large Object

NUMERIC → boolean (1/0), date, datetime

INTEGER small big

REAL → float double

TEXT → char(n) אין יותר מזה אורך המקום varchar(n) אין יותר מזה text (BIG)

Values

NOT NULL: a value is a must, not an option

UNIQUE: a coulom can't have twice the same value (אין יכול להיות לשדה מסוים אותו ערך)

FK PK

Shows

Show_id	title
1	Game of
2	The wither
3	Alf
⋮	

Show id	genre
1	Thriller
123	Thriller
1	Drama
5	Thriller
3	Comney

(☆) SELECT show_id FROM genres WHERE genre = "Thriller"

5
123
1

על שלושתם

תמר לי: "תמרה לי את התוכנית בעלת id=5, בעלת id=1, בעלת id=123"

Nested Queries

SELECT title FROM Shows WHERE id IN (☆) (זה לשמאזיים אחד קצת)

Normality

בעיה: בקובץ הקיבר (אקסל) תחת genre היו כמה values → הפרה של 1NF

פתרון: ליצור טבלה נוספת ולחבר ב-FK

בעיה: יש עדיין תקלה על data. כמה פעמים הופיעה thriller? הוא נופלה הרבה יותר כשה-DB יגדל

פתרון: Shows ← can have many genre ← can have many Shows

Shows — genres

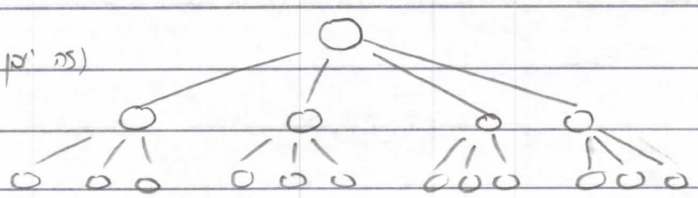
CREATE INDEX title_index ON Shows (title) התיקון יהיה כי זה יותר מהיר

כמה יותר B-Tree

INDEX

CREATE INDEX title_index ON Shows(title)

(Data Structure מבנה נתונים)



B Tree בד"כ

top to bottom: list את שכה עלה, כדי לא לחפש את הנתונים מ-

תיפוש בלי אינדקס לוקח 0.10 שניות אבל אינדקס 0.01 שניות

JOIN

☆ SELECT id FROM people WHERE name = "Steve Carell"

→ 136797 // PK

△ SELECT show_id FROM stars WHERE person_id (☆ ...)

11514

57149 ... // כולה 50 אך ל

Nubers after numbers // כדי לא לחזר כ' כלום

SELECT title FROM shows WHERE id IN (△ .....)

the same thing... but using JOIN

☒ SELECT title FROM people

...→ JOIN stars ON people.id = stars.person_id

...→ JOIN shows ON stars.show_id = show.id

...→ WHERE name = "Steve Carell";

קח את הטבלה X ותחבר אותה עם טבלה Y לפי התנאי.

0.50 שניות

0.05 שניות זה יכול מצוא כל השחקנים ויוצר שבל. אבל כי אין אינדקס צריך לחפש את השחקן

את אבור INDEX יאיזון שלנו (שלא הטבלה לא תגדל אבד) 3 שחקנים → אבור 3 אינדקסים

CREATE INDEX person_index ON stars(person_id)

CREATE INDEX show_index ON stars(show_id)

CREATE INDEX name_index ON people(name);

☒ → I run again → 0.001 second → x50 faster

SQL Injection

a malicious user that insert a code as an input. ex

email adress | DROP TABLE

כאן בדיוק שורצלת סדנת גיבתה ועיתנת גילה יומה דאסב קיתנים

race condition

ex Instagram and likes

מצ'וקצ'קים כקבו קיב .בא גילה זוהי שאנו ← גילה דאסב נתנים ← גילה לכלה שומסת את ה- שאנו

← ++i ← UPDATE

הבעה אתמולה כללנו גולש דומבים פו בכאן עץ שאנו לגוזה היתאנה.

זוהי זוהי לייק ← גילה דאסב ← ספירת מות הלייק (5,000 בויאץ עתון) ← ++i ← 5001 לייק

אחרי זוהי לייק ← " " - " " - " " (5000 בויאץ עתון) ← ++i ← 5001 לייק

↓

5001 לייק בלוים 5002

↓ פתרון

TRANSACTION

Data Structure

CS50

Arrays → limited — array of 3 — not available place — (in memory)

if I want to add another element: I can't

Linked List

the members aren't [1][2][3] in the memory like in array

they are: list_name → 1 (0x123 / 0x456) → 2 (0x456 / 0x789) → 3 (0x789 / NULL) end

they can be anywhere in the memory

that's why we can always add another 1

every value occupy 2 bits: one for the value and the other for the ref of the next value in the list. we allocate twice as much memory for each element

Distadvantage: no Binary Search →?

Advantage: Modified size, flexible

Insert: new → list_name → del1 → del2 → del3

new element enters at the 1st position, not at the last

ב-Linked List אי אפשר ב-pointer כמו במערך לשנות ל' את כל האלמנטים שאחריו

Trees

הקונספט היה שבכל שכבה יהיה פחות אופציות קלפים. אין סיכוי לנצח בקלף שמספרו 3 ומעלה, כל קלף יותר, כדי לנצח

2 Dimension — 4 → 3.5, 6 → 0, 3.7, 5, 10 → 0, 3.5, 3.7, 4, 5, 6, 10

התוצאה

הוספה של אלמנט חדש תהיה מאוזנת (כמעט כלי) עבור ב-2 pointers. כל אלמנט (pointer

(-) For inserting you must do a search

(+) it's ordered → Binary Search

Hash Table

array of Linked List

behave like an array: a → Avi, d → Doron, f → Fadi, g → Gil, h → Hagay, j → Joni

but!

what if I want to add another name that start with a?

behave like an array and like a list: a → Avi → Arno → Adir → Avigayl; d → Doron → Dov → Dori → Dave

Hash Function

it takes as input some string and returns a number

in that case I will like to take a name (like dori) and return a no. that correspond of dori's position (3. why? A=0, B=1, C=2, D=3)

Tries