

Data Structures.

• كل البيانات .

بيانو القيمة لعمل البيانات المطلوب :

الفعالية efficiency ترتبط بـ القيمة من استغلال الموارد
المتعلقة بـ الذاكرة والوقت والجهود والطاقة .



تبديل الرساق context switching \rightarrow مرحلة لإنظام التشغيل متعددة المهام وتنباع بـ تمارك وحدة المعالجة المركزية الواحدة من خلال عمليات متعددة .

جهاز يدعى البرنامج يعاشر (يرتّب) البيانات \rightarrow Garbage Collection \rightarrow يجعل على تحرير مساحة الذاكرة المخصصة للبيانات \rightarrow التي يدعى البرنامج يعاشر (يرتّب) البيانات \rightarrow زمام زمان كل البيانات المختلفة عن طريق فحاصه Time

دالة النمو Growth Function \rightarrow هي دالة تقول إيجاد علاقتها ما بين العمالات \rightarrow الملايين التي تتيح له تنفيذ ويس قيم المدخلات أو عددتها .

$$Y = N$$

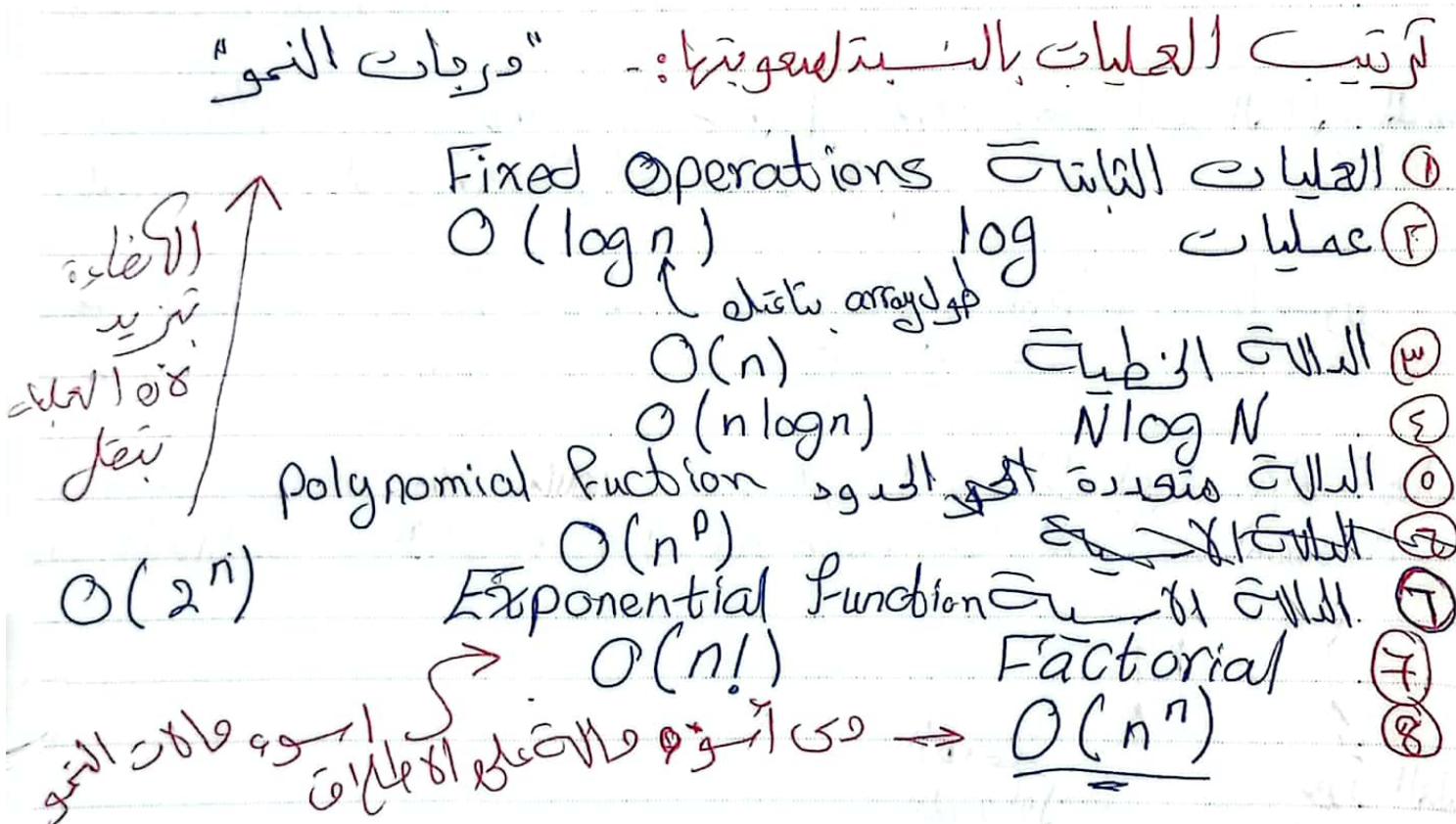
عدد العمالات
التي تتيح

القائمة
التي داخل

النحوينات الأсимптotic \leftarrow Asymptotic Notation
 وقت تدخل الخوارزمي وقت تدخل الخوارزمي مع مدخل معين
 هو مقدار الوقت الذي تستغرقه الخوارزمية مع مدخل معين
 \rightarrow asymptotic notation

$O(n^2)$ \leftarrow هي سعر بيع العقد بـ n^2 معاولات النحو
 Worst Case Scenario

Best Case scenario $O(1)$ (omega - 2)
 Worst Case scenario $O(n)$ Big O
 average case $O(n/2)$ (theta Θ)
 $\downarrow \equiv (\Theta = n)$ كلغ العقد بـ n



نحوه و ترتيبه يخوا Array

Array in Java → ① تعريف المفهوم ② إنشاء المفهوم
in Java :-

int[] numbers = int []numbers = int numbers[]

→ int[] numbers = new int [10]

↳ Keyword

يُ alok. memory لـ array out go ↳

0 0 0 0 0 0 0 0 0 0

address لـ location memory لـ كفر لـ جاء ↳ Java

Windows

32 bits

main difference
in addressing

Windows

64 bits

01 ————— | 2^{32}

$0 \rightarrow 2^{32} - 1$

02 ————— | 2^{64}

$2^{60} \rightarrow$ Petabytes

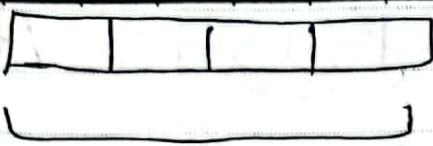
2^{10} = Kilobytes

2^{20} = Megabytes

2^{30} = Gigabytes

4 GB RAM

Integer



4 bytes (32 bits) = 4 addresses (العنوان)

`int[] arr = new int[10]`

لقد أنشئت متغير arr من النوع int يحتوي على 10 مسارات من العنوان addresses باربعين byte (40)

Random access → الوصول العشوائي access = الوصول العشوائي array

Operations and time complexity in arrays:-

① عمليات إنشاء وتحريك array ← array بسرعة جداً (أقل من 10ms) (الإضافة والremoval)

Time Complexity $O(1)$

وتحريك array ← عمليات إنشاء وتحريك array (Default ب速度快) (shallow copy) ← وتحريك المثلث

→ $O(n)$ ← array (إضافة كل العناصر) ← المغادرة مع تهيئة كل العناصر

③ عملية البحث ← array (البحث في كل العناصر) ← قراءة كل عنصر ولكن نعم

Search or linear search

→ $O(n)$ ← عملية ترتيب العناصر من الأكبر للأصغر أو العكس

④ عملية إدخال وال удал ← array (الإدخال وال удал) ← دمج 2 نodes في 1 node واحد يحتوي على خطوة

وسيكون array [1 2 6 5] ← لو عايز تضيف رقم بين 2,6

يحتوي على 5 بعدين إلى 6 مكان - يبقى فارغ بعدين تضيف المكان

Time Complexity $O(n)$

Deep copy

- ~~الكل ينادي المبرمج~~
- ~~يطلب المبرمج~~
- ~~لطبع المحتوى~~
- ~~عنوان الـ address~~
- ~~من من اـ areas~~

Complex Variables

objects → class

properties and methods

Shallow copy

طبع المحتوى
عنوان عاوزه

Primitive Variables
int, float, boolean

Shallow copy

Car1

Car2

Car object

Red

speed

automatic

لـ copy address يـ copy object . فقط يـ copy object

فـ copy object فيـ copy array . فـ copy object فيـ copy array

دـ المـ copy المـ copy المـ copy المـ copy

عـ على . (مـ copy Deep Copy لـ copy array . copy المـ copy المـ copy المـ copy المـ copy)

ربـ المـ copy المـ copy

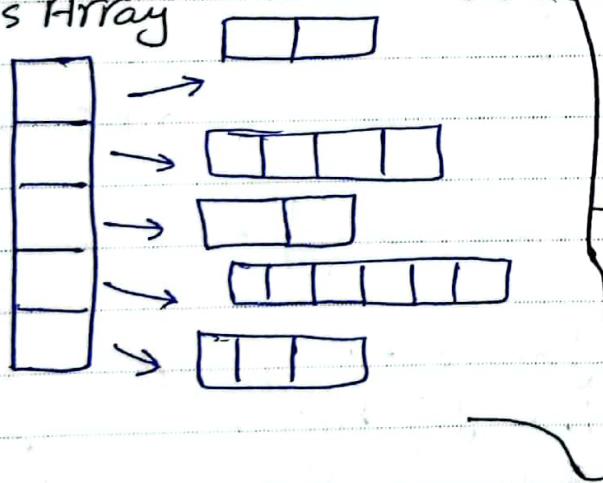
. TWO Dimensional Array .

لكل معايير : دلوقتن افنا قولنا إن ال array يقدر تخزن
 نوعين من القيم ① primitive أولاً ←
 الباقي object يخاور على ال references for objects ②
 بناءً على address ال دلالة في المخزن

و بالذال نقدر نعيّن له خاصية object ← array ←
 معناته ليه و ممكن تخزينه وال آخرة وبطبيعة ونهاية ←
 object يقدر اخبار ال reference تابعه ←
 و طبعاً إن ال array يخاور على arrays ←
 creates the array ←

Array of Arrays :-

Rows Array



Columns Arrays

وطبعاً أصبح ال array له
 two dimension
 أقدر اسميها كده
 "Two Dimensional Array"
 "جامعة البحرين"

arrays ازای اعرف بالنسبة لـ two-dimensional-array (جای ازای ازای)

`Int [][] arr = new int [][];`

↓ این که بخواهد گزینه توانی بگوییم

که array را بتخواهیم داشت الی Compiler نمایش two-dimensional-array (پس از array of array) را بخواهد.

وقبلاً line by initialization یا اینها هستند.

① 2D array with fixed length for all Columns:

الا عددية الصيغة:

`int [][] arr = new int [10][5];`

لهم چنانکه در نظر داشتم که می خواهد در array دو عدد خاص را بخواهد و در این دو عدد این که این اول array بتخواهیم داشت که یعنی باعث شود این نفر از طول و عرض آن بخواهد باشد و خواهد داشت ۵۰ میلیمتر

② 2D array with variable length for each column:

که این ایجاد کنیم که این متغیر که در نظر داشتم که اینها برای مجموعه ای از arrays می خواهیم که اینها دارای طبعات متفاوت باشند اینها می خواهیم بخواهیم که اینها برای مجموعه ای از objects می خواهیم که اینها می خواهیم که اینها برای مجموعه ای از objects می خواهیم

- می خواهیم اینها برای مجموعه ای از objects می خواهیم که اینها برای مجموعه ای از objects می خواهیم

لهم که اینها برای مجموعه ای از objects می خواهیم که اینها برای مجموعه ای از objects می خواهیم

ex. `int [][] numbers = new int [10][];`

numbers = new int [5];

①

②

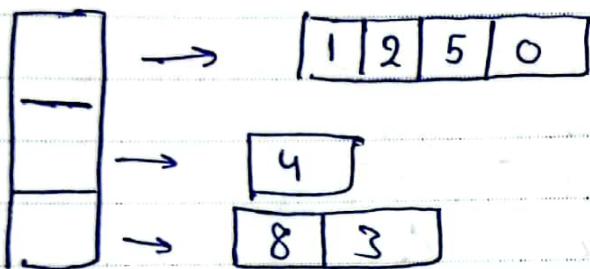
③ 2D array with initial Values :-

ومن الطريقة التي يُعرف الـ array بقيم من البيانات

`int arr = {{1, 2, 5, 0}, {4}, {8, 3}}`

Curly brackets

ref. على سُرُّ array يعني أن هذه القيم هي قيم array



Operations on 2D Array.

لِمَنْ تَعْتَدُ الْمُصْوَلُ لِلْعِصْرِ فِي
أَرْبَعَةِ قَدَرٍ قَيْمَسَ عَنْهُ تَوْصِلُ مُلْكَانَ

ex. `arr[2][3]`

العنصر الثالث والرابع باذن العنصر
~~pointer~~ ← لـ array باذن العنصر

لـ استرجاع كل عنصر الـ

عن طريق كِبْ لِيْ مِنْ وَادِي بِرْجِ
وَادِي دِي لِوْجِ لِكِنْ أَشْنَ لِكِنْ مُرْجِ
بِلْ لِكِنْ الْعَدَدَةَ (الْعُصْرِ لِغَيْرِهِ) وَ الـ loop
عَلَى الصُّفُوفِ بِتَابِعَتِ الـ arrays اِنْ بِسَارُ عَلَيْهِ

```

ex. For (int i=0 ; i<arr.length ; i++) {
    For (int j=0 ; j<arr[i].length ; j++) {
        System.out.println (arr[i][j]);
    }
}

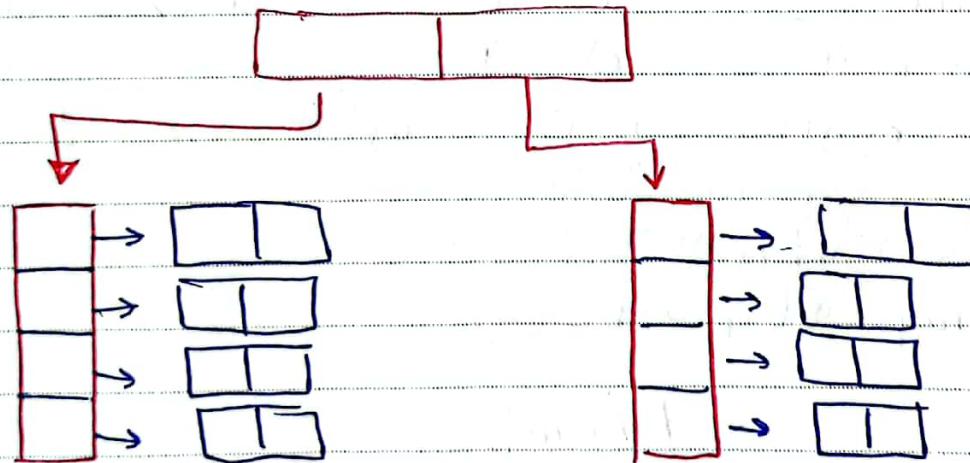
```

Time complexity $\rightarrow O(M * N)$

(3D Array) Multidimensional Arrays

بساطة جداً في 2D array هي من "بخار" object و عبارة عن array of array يعني مفهوم مماثل في 3D array

ex.



\rightarrow int [[[]]] numbers = new int [2] [4] [2]

لـ كل اول رقم يزيد عن 0 يعني في الـ memory فـ

الـ array عبارة عن مجموعة من العناصر اما ان array تكون مكونة من primitive types او objects ذاته او تكون الـ array مكونة من objects ذاته وناتج عن ذلك.

• ArrayLists المصنفون الديناميكيون.

دلوقة أنا لو بعمل To-do-list أو user بدل كل قام غير فكده معنى اغير مساعدة أكثر أو أصغر من اللي بالـ user محتاجها فكده الممكنة دي تحل ازاي؟

بسطة عدى option في معظم لغات البرمجة يقدر بعد خصائص الـ array ومعها خصائص الـ list وعذراً ما زلت مكان صدر والذاكرة هي الأول.

ArrayList Class Java هو الـ option

هو مثيلات معرفات عدد عناصر array (Pg) ولكن يجب تأسيب عنصر منه كي يتبعها بسرعة جداً فذلك عارف ① ترتيب العنصر مكان أول عنصر في الذاكرة ② عدد العناصر ③ وهي قائمة متداولة بعدها عارفها زي هنالك أو To-do-list اللي ذكرناه فوق عناه كده

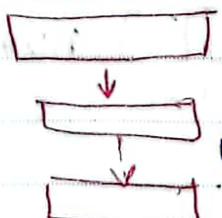
لـ ArrayList



كده واحد كل خصائص الـ array

ArrayList

List



كمواحد فخصائص الـ list وهو مكون من الملفات تزيد أو تقل عما في الـ array.

جُنْدِي لِلْجَمِيعِ الْجَمِيعِ

Java - الـ ArrayList

- ① `ArrayList<Integer> myList = new ArrayList<>();`
الـ myList هي مساحة لـ ArrayList فما هي المفهوم؟
- ② `ArrayList<Integer> myList = new ArrayList<>(10);`
[10] = مساحة لـ ArrayList
- ③ `ArrayList<Integer> myList = new ArrayList<>(Arrays.asList(new integer(2), new integer(3), new integer(8)));`
الـ myList هو المثلث عاليه بـ 30 درجة و المساحة من المثلث
array list هي نسخة Shallow copy و مطابقة بينها وبين list
وليس مثل array بل ما تعلم في الـ array فـ list يختلف عن الـ array

ArrayList $\xrightarrow{\text{لا يقبل غير}}$ Objects $\xleftarrow{\text{كلها}} \text{List}$
 primitive $\xrightarrow{\text{غير}} \text{Object wrapper}$ $\xrightarrow{\text{كانت ابتداء}}$ و لكن اقدر بطريقة كانت ابتداء

Eg. int $\xrightarrow{\text{Wrapper class}} \text{Integer}$

ArrayList operations) → ArrayList العمليات على القائمة
arraylist.get(2) تحتاج إلى إدخال فرط طابع
arraylist.set(2, 100) تحتاج إلى إدخال فرط طابع

- اسْتِرْجَاعُ عَيْنِرْ : ①

arr[2]; ← square brackets (괄호) of array فَوْلَ

↳ 1 Function مُفْعَلَة ← ArrayList جَهْد
ex. arrlist.get(2); • get

قيمة : ⑤
- : arrlist.get(2)

arr[2]=100; ← العتبة التي تغير العنصر array فَوْلَ

وَعْدَتْ set بِـ+ Pun. مُفْعَلَة array List جَهْد
↳ get

ex. arrlist.set(2, new Integer(100));
 Index العنصر الذي يخزنها ورمز ↳
 كونه من نفس النوع

- اسْتِرْجَاعُ عَيْنِرْ : ④

System.out.println(arr.length) array فَوْلَ
↳ array ما هو اسم

System.out.println(arrlist.size()); arraylist فَوْلَ
↳ arrlist ما هو اسم

ـ : arrlist مفهوم الإضافة والremoval

عن طريق استخدام الـ add, remove

ex. arrList.add (new Integer (10))

ـ القائمة هي قيم تسلقية وآخر الـ arrlist

arrList.add (2, new Integer (10));

ـ مكان العنصر الذي يُضاف فيه

ـ القائمة

ـ arrlist.add(index) ينبع من العنصر الذي يُضاف في المكان الذي يُحدد في index

ـ يساوي 2

ـ مفهوم المدفون :

ـ → arrList.remove (2);

ـ يعني إزالة العنصر الذي هو arrlist.get(2)

ـ → arrList.remove (Object);

ـ هنا إنما تشير إلى أول مرتاح فقط arrlist.get(0) هو موجود فيها وهو عادي جداً

ـ arrlist في المنهج

ـ (ـ جمع كل الأشياء وهم قبل كده

ـ fun. clone *ـ

ـ لاحظ إنها تحصل براضا عليه و ما يعنى arrlist.clone()

ـ deep copy فالطريق مفترض هنا الأشياء وain't primitives option هي التي تخرج objects

ـ → ArrayList<Integer> newList = arrList.clone()

Time Complexity with ArrayLists :-

① عمليات السترينج وتحريكه سهلة ← get , .set
 لعمليات تغير أو استرجاع القيم → O(1) ← مره واحد على كل طول

O(n) ← For loops ②
 لـ arr ← في بعدي عناصر arr

O(n) ← O(1) * O(n) ← Clone ③ النسخ

④ عمليات الإضافة أو الحذف من
 O(n) ← adding ← remove ← و الحذف
 المتصفح

⑤ في حالات الاختلاف من الآخر ← كمزح عالي

← أفهم دره معياراً احنا قولنا ان arrList يكون خاص ، والكل و
 أو ما تدخل قيمة بمحض مكان لها و ~~وهو~~ يزداد هنا إمكان [1.5]
 من الوضع الذي كان عليه سابقاً
 يعني لو هو موجود ولو قررت على إمكان وفي مكان خاص حيث
 ذي عنصر براقتله طب لومفيت 0 ؟ فتروج arrlist وانه
 مكان آخر في 15 خانة وتنفع arrlist التي كانوا موجودين في
 الأول وتسأله الحسن إمكان تزوده فيما براقتله يعني كده عند الناسين.

⑥ تحصل عمليات الاختلاف مباشرة
 يتم زيادة المساحة أول ونفع العناصر ثم الاختلاف ⑦

طلب كده هنا خارج من O(1) و O(n)

اصل إننا كناد المتبسط و هو مصطلح جديد

$$\text{Amortized Time Complexity} = \frac{\text{إجمالي عدد عمليات النسخ وإضافاته}}{\text{عدد عمليات الإضافات}}$$

مثال ← لو اعتننا إتنا حسناً بـ $\Theta(n)$ واحد وكل ما اجي افرز عنصر لا مفتش سـ $\Theta(n)$ هيدل نسخ و الاماكن ضئيلة الضغط يعني كده عشان أوهيل دـ 128 عشان فعل نسخ گام موه ۳ حلاته

$$\text{عمليات الإضافات} = 128 \\ 127 = 64 + 32 + 16 + 8 + 4 + 2 + 1 = \text{"النسخ}$$

$$\frac{2}{128} \approx \frac{(127 + 128)}{128} = \text{Time Complexity}$$

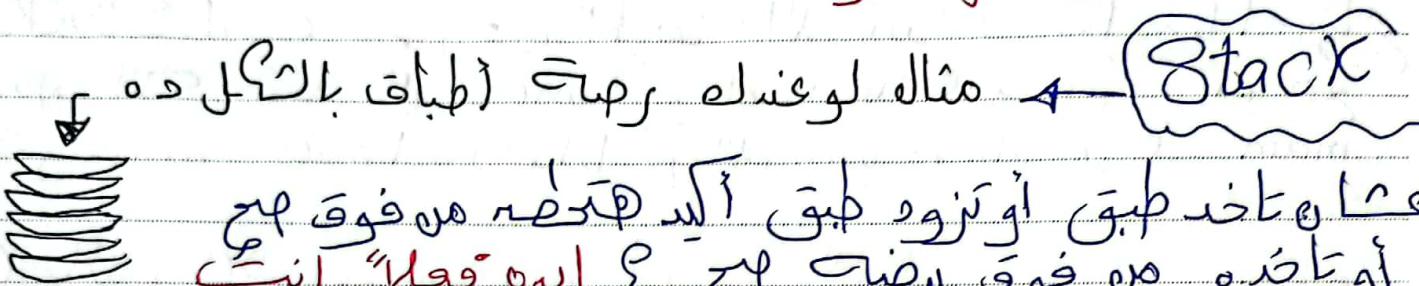
$O(1)$ ← Time Complexity °°
و نـ بـ تدخل عمليتين فقط نسخ وإضافات وبالرـ الـ عدد ثابت عـ الـ طـول.

stack الرسم

لاحظ ان المهم قبل اي حاجة و مع البيانات ازاي من الذكرة و يبقى عارف العناصر دى هر فئومت او منرتبا ازاي.

يعنى السلوكيه اللي انتـ بـ تعامل سـ مع اللـ العـناـصر و هي كلـها هو اللي سـعـنـرـ لهـ لـانـ اـنتـ الليـ بـتخـارـ الطـرـيقـاتـ الليـ هـتـنـفـلـ بـها علىـ الـبيانـاتـ دـى.

بعد ما عرفنا كل المعلومات السابقة فنطلع ليقل
أعلى وسيلة إننا ولو قمنا من ضمن لفيف البيانات على طريقة
وجوده والذاكرة زى الـ array, arrlist بل طريقة المtower
أو القبود على ~~فيف~~ البيانات المطلوب استخد اهم وليس
طريقة وجوده والذاكرة



Data Structure

LIFO Last In First Out
آن ال الأول وهو اللي يعن ال Stack ويس
آخر حاجة خطيرها هي أول حاجة
باتقدما

لدي بالذات مكان افراد طبق من حيث لازم اعدى على كل الأطباق
والطريقة هي اسمها Sequential Access يعني استرجاع
متتابع من هرمه وأوله
افتقر مثاباً لأن كنت أقدر ابي أي عنصر خال
Random Access من غير ما اعدى على كل العناصر وده ممتاز
حال الطول

طبقة زوال : ليه كل القبود على ال
لتخدم array و ترجع دماغاته؟

القبود مهم لكنه عازز توفر الـ User المعاشر من العالم
اللى هو يحتاج فقط لغير مكان تعرف تزود له كفاءة المنتج
وهو فيه اهمية Abstraction

من خلال عملية التجزيـة نـظر المـبرمج جميع البيانات المـتعلقة بالـبيانـات مـاعـد الـبيانـات ذاتـ الصـلة (الـTime Com. user مـحتاجـها) من أـجل تـقليل التـعمـق (الـTime وـزيـادة الـاستـادـة).

Ex. on Stack:-

وـهي الطـرـيقـة الـسـادـى لـتـنـفـيل بـهـا إـيـ برنـاجـ وـهو إـنـه يـنـفذ كـلـ Func. وـرـاهـ الـفـانـيـات وـلـهـامـ جـمعـ بـسـجـعـ main خـارـجـهـ نـفـذـهـا وـيـسـلـهاـ تـمـ الـقـبـلـهـ لـهـماـ وـصـلـ لـ الـ

أـنـ الـأـنـجـ وـهـوـ اـنـ تـسـتـخدـمـ عـالـظـلـ XML, HTML < Markup Language (R) < وـ لـفـاتـ الـتـمـيزـ

<div> <div> تـرـسيـلـ لـقـلـ وـقـعـ →
<p> <p> ✓ جـوـهـارـ Tag
</div> </p> بـتـاعـهـاـ مـشـ SCOPE
</p> خـارـجـهـ وـفـوـ نـفـنـ اـسـتـخدـ </div> الـ stack (الـ اللـوـكـ يـغـرـزـ الـ طـرـيقـةـ بـنـاعـتـ الـ stack)

Stack Definition In Java:-

① Stack < Integer > mystack = new Stack<>();
ArrayList < Integer > myList = new ArrayList<>();

لـوـاحـظـتـ هـوـ مـنـ وـافـدـ فـيـ اـسـنـانـ لـهـنـ يـغـرـمـ معـ الـوقـتـ
عـلـىـ حـبـقـ يـسـتـخدـمـ وـإـنـ وـبـرـضـهـ هـوـ يـتـخـدـ مـ بـسـ مـشـ قـيمـ primitive

integer() → int (x)

لـوـجـيـزـهـ مـاـقـولـاـ int الـ wrapper class

طبعاً يضيف كل (push) عنصر في خلف stack ← Fun. Push ②

وادر فوق لآخر (last) ← myStack.push(new Integer(20));

آخر عنصر ← Fun. Pop ← استرجاع ③

Integer lastElement = myStack.pop();

ويحفظ لواحد تخدمه pop كرمه مبادرة في loop ←
أيضاً "أين" صابة لكن نوعها تعرف قيمته بمن فتخدم صابات
كذلك.

آخر عنصر ← قيمته آخر ← peek ④

stack.size() ← طرفية عدد العناصر الموجودة داخل الـ stack ⑤

stack.isEmpty() ← طرفية هل الـ stack فارغ و لا ← ⑥

لتحديد ما إذا كان العنصر موجود وليس قيمة ← search ⑦

Stacks and Time Complexity.

الـ Java : متكرر ← الـ stack ; المتكرر ← الـ built-in func.

Stack inherit ← Vector

inheriting from Vector ← يعني يورث من Vector خصائصه وطريقاته
خزينة العناصر والعمليات التي تجري على خزن

للب دیس ار جیئن بیانات جیئن
 ↪ Vector
 ↪ arrlist افضل ناله ٿيں استاد ام
 ↪ arrlist جگہ arrlist بکثرة

ArrayList = ↪ Vector

ArrayList = ↪ Stack stack اور خروجیos
 arrlist اور بطریقے خروجی memory اور

$O(1)$ ← POP , Push ڈیلیس ①

لپڑا عبارت اور سطح اور سترجع عبارت وار فریط
 اور اچھا کیا جائے اور اچھا کیا جائے اور فریط

$O(1)$ ← Peek ②

$O(1)$ ← IS Empty ③

دیکھو کا

$O(1)$ stack operations ایک Time Compl. ایک
 لپڑا وقت کیس و پھر علیکه وقت کیس
 لو اس تینوں میں

الطاور Queue .

→ A Queue is defined as a liner datastructure that is open at both ends and operations are performed in First In First Out (FIFO) order

← لعن قو فكيل بيانات ت خطمه من الطرفين FIFO و يتم تنفيذ العمليات

نحوه على نوع اسماز بدلار queue

ابوه ينفع و لعن مدن خفاف القوو الملي على queue بعدين و في arrlist تقدر تضيف عيرون الى تبقي و كده انتي بتحت العرف اذا ستن الى اتجعل عيانت او arrlist

لخط ان ال queue خوار arrlist بعدين انت معرفت من class من queue

أواي بس اور ٩٩٠٥ interface و ايه الفرق بين ٩٩٠٥ class وبين اور

class من object بس اور انت معرفت ان اقول لك ان ٩٩٠٥ interface بعدين object او معين

Difference between class & Interface

Class (الفن) → A class represents the set of properties or methods that are common to all objects of one type.

هو الآخر لأن ال class وهو مطبوب حابات بخرجاته objects لها نفس الكيل والمفهون مع افتراض بعد ال القيمة values

مثال : "العربيات" ← لها تكون من Components عبارات الفرنسية و إنكروشات و الزجاج و المواسير بمح !

له تبع العربيات هو ال Class بتاعت قدرت لدينا objects لس نعمل نفس الحاية وهي إزها توصله من مكان و لكن تختلف في بعض الخصائص مثل عربيات BMW ، عربيات manual ،

طب ال interface (واجهة) عبارة عن أي

Kelvin مثلاً إنه معالم مفتح ولو قرئ ممكن يفتح بباب أيه ؟ مثلاً باب او ضوء او باب شقة ولو مفتوح صغير ممكن تفتح بباب قفل أو خزانة لكن قبل ممكن تفتح بباب جاك مثلاً أو بوتجاز ؟ متأجيل لأن أصلها مفتوح يمكن مفتوح من الأساس .

افهم منه أنه كل اللي خرق دخل اشتراكوا في مابس و هو إن لهم مكان لدخول المفتاح اللي هو ال interface وهو طريقته معينة اشتراك فيها انه . يفتحوا بالمفتاح

مع آخرها و ما فيهم

لقد انا اد class اقدر استخدم مع objects تيرجيم
لهم نفس الاعمال المترددة

لأن الـ interface يعبر شرطها وأوامرها من class يلزم
بها وساعتها ان class يفهم ان الـ class هو . ينفذ
هذا object على interface وظائف .
وخلال أول ما يفهم Compiler يعرف أنه قروء بجرا
built in interface على تفاصيل أوامر الـ class تكون
في اللغة التي انت شغالة بها ولو فيه امر واحد ناقص اد Compiler
هذا ينفذ الى الـ class عاوزه

مثال : لو عندك loop على stack وqueue وarray
ومن هندر حما اي الـ loop ؟

في الـ Java معن interface له تسلقها ~~لكل~~
 Iterable Interface و ~~Iterateable~~ ~~for~~
 ان البيانات في loop تدخل عليها وتنفذ برضاه
 وهو كل من الـ loop عاشه تسلقها ~~لكل~~
 iterator interface سعادتها تقدر تستخدم ~~لكل~~
 next value
 بساعتها
 ~~for~~

queue in Java

الانترفيس في الـ Java يتبع الـ queue
Func. يحتوى على الـ push .

لدخول العنصر للطابور Enqueue Function ①
خروج العنصر من الطابور Dequeue Function ②
اظطر لبيان الطابور Function ③

و في نوعين من الـ Func دى و حالات صورت

Exception ١. اول واحد سريع null ٢. سريع.

Fuction

Exception

null

Enqueue

add()

offer()

Dequeue

remove()

poll()

Top of queue

element()

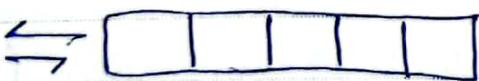
peek()

Queue الـ interface لـ Java او C# بـ (Double-Ended queue) Deque بطرق مختلفة منها

في Priority queueing و هو الـ خاص من خارج و را يجنب لا بل فتح جميع المفهومات باعترافها.

نظام دخل على Time Complexity و لكن Queue الـ Time Complexity انا معروض ابيه له لوحة عن خرافة implementation class عامل عالي بين و لكن هي حبـ عـ طـرـيق الـ Queue الى انا عاوزه و امثال الى يستعمل عليه هو Array Deque class

Deque → (Double Ended queue) → يعنـ اـ تـ وـ فـةـ → يعنـ تـ قـدرـ دـخـلـ وـ خـرـجـ عـنـهـ مـنـ النـاحـيـتـ وـ مـالـلـ يـقـدـرـ دـخـلـ لـ Queue stack اـ عـمـاـنـ مـكـنـ اـ خـرـجـ مـنـ نـاقـيـةـ وـ اـ دـفـلـ مـنـ نـفـرـ النـاحـيـتـ بـ رقمـ لـ كـمـ اـ حـتـاـهـ تـحـدـيـدـ دـلـوقـيـ



طريقة عمل الـ Array Deque

Array Deque $\xrightarrow[\text{interface}]{\text{implement}}$ Deque $\xrightarrow[\text{interface}]{\text{implement}}$ Queue

لحوظة (١)

عملية المضافة ← ← ←

كرنفال الكورس قبلي
خاص -

(dequeue)

عملية اخراج ← ← ←

دلوقي انت لما تدخل dequeue \rightarrow [] \rightarrow
اطرفاعي بسب مكانه فاضي صبح وغدا
كمل فرزخ باقى العين عينه يقعوا مكانه
و بالتأل ال Time Comp. (n) O و ده سرع
جد "حدا" و أكيد فيه حل!

الحل هو إنا نعمل مؤشر للـ Dequeue والـ Enqueue ← ← ←

لما

↓ Dequeue

أيا طلب و بعدين للأول لآخر فنصر

↓ Enqueue

عندي و عاشر ازود في جم الـ array

↓ Enqueue

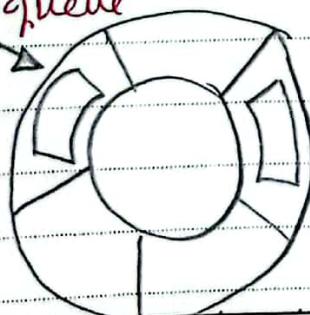
بعنوان الـ array له معروض ازود فيها عاشر

↓ Enqueue

و بالتأل لازم ازود مساحة الـ array و كده فتعمق
ان عاشر اماكن فاضيتس كشن.

طب اهل إينا هنعمل الـ Circular array ← ← ←

Enqueue



عنه اعرف أصنف واحدني، رافق من غير ما ينفعون

فيه بدلاته ونهايات الـ array ودهن بجعل زيارته

لسادسة غير لما تصل خالص وده بمعنفه

↓ Dequeue

لما بتلاقي مؤشر الـ dequeue والـ enqueue وهو مؤشر الـ

وراء بعض

الإطارات الـ $O(n)$ في التحالف

Array Deque

و $O(1)$ للخطاب والرد و $O(n)$ Time Complexity

-: queue لـ خدامات

Round Robin هو نظام التبديل المموج باسم الدور ①
وهو يعتمد على queue ②
 ③ (يكتسب كل طلب الدفعي)

Breadth First Search خوارزمية انتشارية ④

وظائف واجهة