LECTURE 01

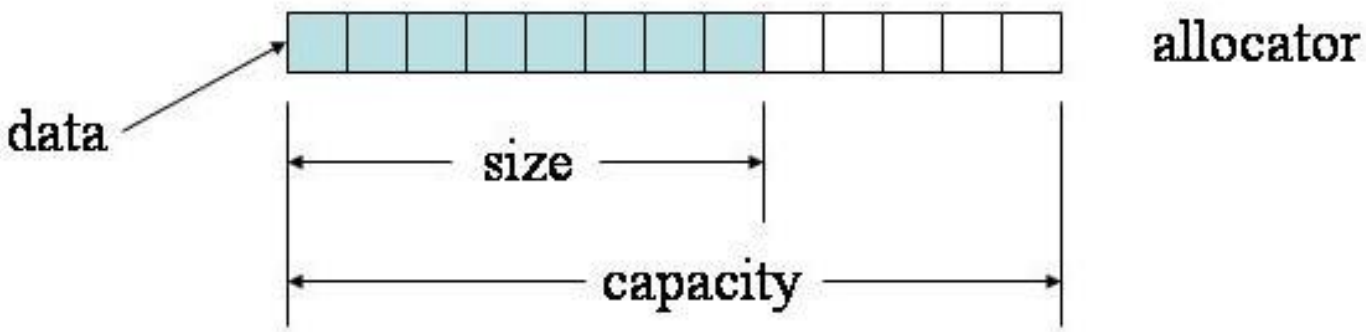
DYNAMIC ARRAY & STRING

Phạm Nguyễn Sơn Tùng

Email: sontungtn@gmail.com

Dynamic Array là gì?

Dynamic Array (Mảng động) là cấu trúc dữ liệu dùng để lưu trữ các đối tượng có kích thước không xác định. Khác với mảng tĩnh, người dùng không cần phải khai báo trước số lượng phần tử khi sử dụng.



C++: **vector**

Java: **ArrayList**

Python: **list**

Cách khai báo sử dụng Dynamic Array

Thư viện:

```
#include <vector>
using namespace std;
```

Khai báo:

```
vector<data_type> variable;
```

```
vector<int> v;
```

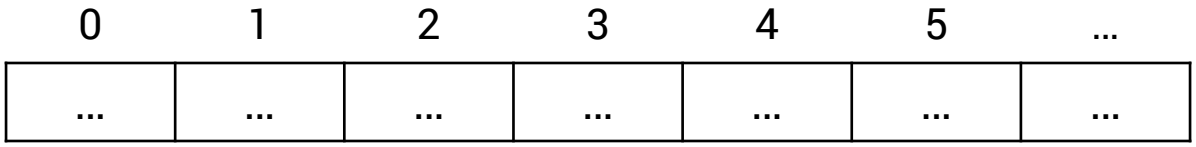
Thư viện:

```
<không cần khai báo thư viện>
```

Khai báo & khởi tạo:

```
variable = [value1, value2,...]
```

```
l = []
```



Cách khai báo sử dụng Dynamic Array

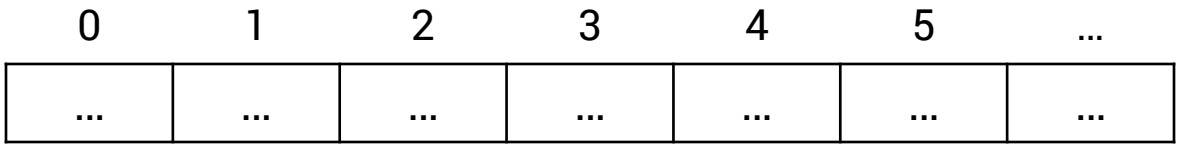
Thư viện:

```
import java.util.ArrayList;
```

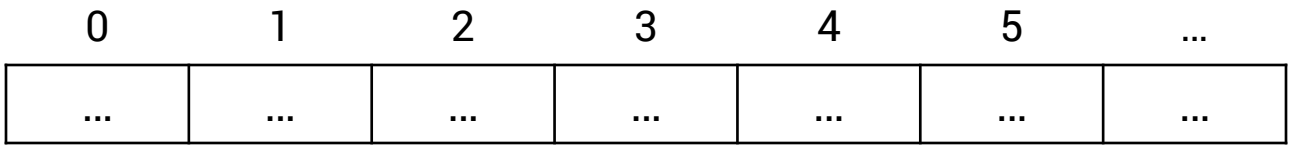
Khai báo:

```
ArrayList<Class> variable = new ArrayList<Class>();
```

```
ArrayList<Integer> a = new ArrayList<Integer>();
```



Thêm phần tử vào Dynamic Array



C++

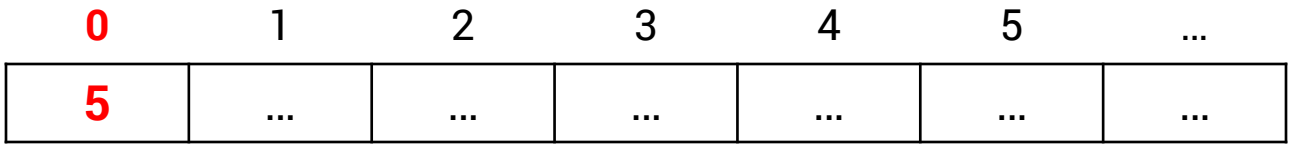
python™

push_back(value)

```
vector<int> v;
v.push_back(5);
```

append(obj)

```
l = []
l.append(5)
```



Lưu ý: các phần tử được thêm vào lần lượt vào cuối mảng.

Thêm phần tử vào Dynamic Array

0	1	2	3	4	5	...
...	...	...	...	...	...	...

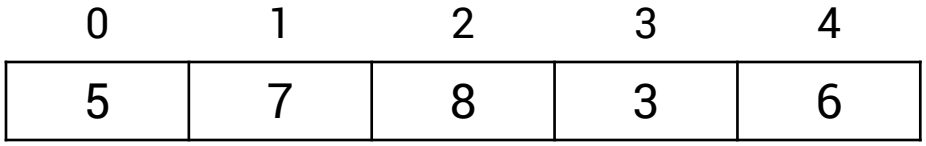
add(element)

```
ArrayList<Integer> a = new ArrayList<Integer>();
a.add(5);
```

0	1	2	3	4	5	...
5	...	...	...	...	...	...

Lưu ý: các phần tử được thêm lần lượt vào cuối mảng.

Chèn giá trị vào Dynamic Array

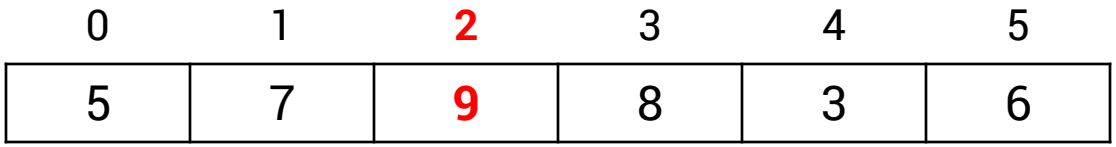


insert(iterator, val): Chèn giá trị val vào vị trí iterator.

```
vector<int>::iterator it;
it = v.begin() + 2;
v.insert(it, 9);
```

insert(pos, obj): chèn giá trị obj vào vị trí pos.

```
l.insert(2, 9)
```



Chèn giá trị vào Dynamic Array

0	1	2	3	4
5	7	8	3	6

add(index, element): Sử dụng lại hàm add có thêm một tham số thứ 2 để chèn một phần tử vào **vị trí index** của ArrayList. Throw exception nếu index không thỏa mãn giới hạn của ArrayList.

```
a.add(2, 9);
```

0	1	**2**	3	4	5
5	7	**9**	8	3	6

Lấy kích thước của Dynamic Array

0	1	2	3	4
5	7	8	3	6

size()

```
int n = v.size();
cout << n;
```

len(obj)

```
n = len(l)
print(n)
```

5

Lấy kích thước của Dynamic Array

0	1	2	3	4
5	7	8	3	6

size()

```
int n = a.size();
System.out.println(n)
```

5

Truy cập phần tử trong Dynamic Array

0	1	2	3	4
5	7	8	3	6

v[index]: Giống như mảng tĩnh ta có thể dùng index dùng index để truy xuất giá trị tại một vị trí bất kỳ trong mảng động (0 ≤ index < size).

```
int result = v[2]
cout << result;
```

8

Truy cập để thay đổi giá trị.

```
v[4] = 9
```

0	1	2	3	4
5	7	8	3	9

Truy cập phần tử trong Dynamic Array

0	1	2	3	4
5	7	8	3	6

bonus

front(): dùng để lấy phần tử đầu tiên của mảng.

```
int result = v.front();
cout << result;
```

5

back(): dùng để lấy phần tử cuối cùng của mảng.

```
int result = v.back();
cout << result;
```

6

Truy cập phần tử trong Dynamic Array

0	1	2	3	4
5	7	8	3	6

l[index]: Giống như mảng tĩnh ta có thể dùng index dùng index để truy xuất giá trị tại một vị trí bất kỳ trong mảng động (0 ≤ index < size).

```
result = l[2]
print(result)
```

8

Truy cập để thay đổi giá trị (có thể dùng index -1 hoặc index 4)

```
l[-1] = 9
```

0	1	2	3	4
5	7	8	3	9

Truy cập phần tử trong Dynamic Array

0	1	2	3	4
5	7	8	3	6

get(index): Giống như mảng tĩnh ta có thể dùng index dùng index để truy xuất giá trị tại một vị trí bất kỳ trong mảng động (0 ≤ index < size).

```
int value = a.get(2);
System.out.println(value)
```

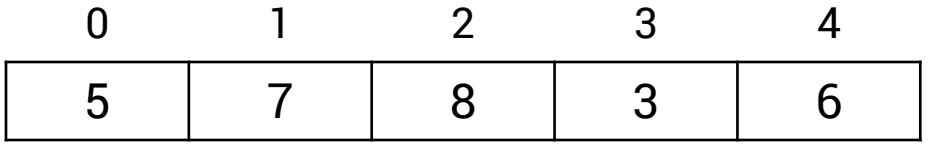
8

set(index, element): Truy cập để thay đổi giá trị.

```
a.set(4, 9);
```

0	1	2	3	4
5	7	8	3	9

Xóa phần tử cuối khỏi Dynamic Array



C++

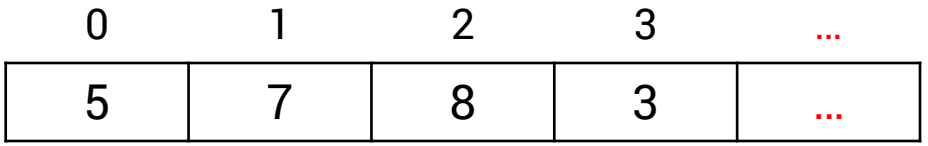
pop_back()

```
v.pop_back();
```

python™

pop()

```
l.pop()
```



Xóa phần tử cuối khỏi Dynamic Array

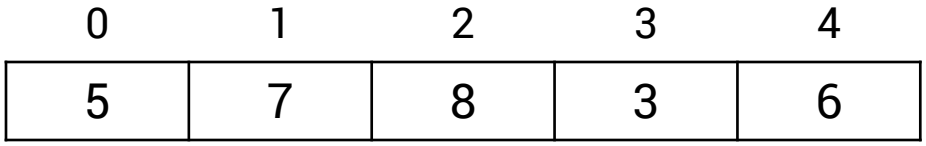
0	1	2	3	4
5	7	8	3	6

remove(index): Để xóa phần tử cuối cùng của ArrayList, ta sử dụng hàm remove để xóa. Lưu ý chương trình sẽ Throw exception nếu index không thỏa mãn giới hạn của ArrayList.

```
a.remove(a.size()-1);
```

0	1	2	3	...
5	7	8	3	...

Xóa phần tử ở vị trí bất kỳ trong Dynamic Array

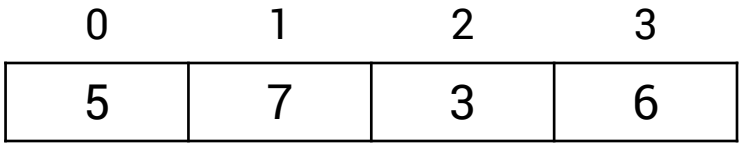


erase(iterator): Xóa **một** giá trị.

```
vector<int>::iterator it;
it = v.begin() + 2;
v.erase(it);
```

pop(pos): Xóa **giá trị** trong list ở vị trí bất kì và trả về giá trị bị xóa.

```
l.pop(2)
```



Xóa phần tử ở vị trí bất kỳ trong Dynamic Array

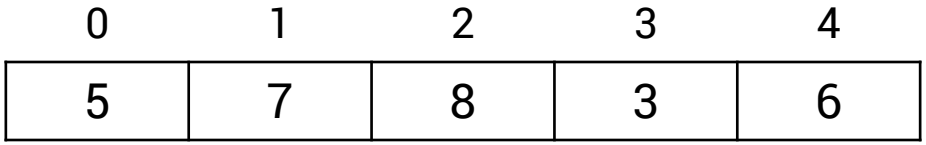
0	1	2	3	4
5	7	8	3	6

Dùng lại hàm remove để xóa phần tử bất kỳ trong Array List, sử dụng thêm tham số index **remove(index).**

```
a.remove(2);
```

0	1	2	3
5	7	3	6

Xóa toàn bộ các phần tử trong Dynamic Array



C++

clear()

```
v.clear();
```

python

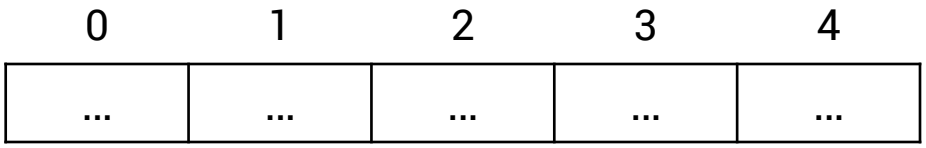
clear()

```
l.clear()
```

Java

clear()

```
a.clear();
```



Tăng kích thước của Dynamic Array

0	1	2	3	4
5	7	8	3	6

resize(size): Thay đổi và gán giá trị bằng giá trị mặc định của kiểu dữ liệu **nếu có thể**.

```
v.resize(7);
```

extend(list): Để tăng kích thước của list, ta nối thêm list mới vào list hiện tại.

```
l.extend(2*[0])
```

0	1	2	3	4	5	6
5	7	8	3	6	0	0

Tăng kích thước của Dynamic Array

0	1	2	3	4
5	7	8	3	6

Java không có hàm resize mà phải tự cài đặt lại bằng cách dùng hàm add.

```
for (int i = 5; i < 7; i++) {
    a.add(0);
}
```

0	1	2	3	4	5	6
5	7	8	3	6	**0**	**0**

Giảm kích thước Dynamic Array

0	1	2	3	4
5	7	8	3	6

C++

```
v.resize(2);
```

python

Để giảm kích thước của mảng trong python, ta sử dụng slicing để (cắt một phần của mảng) và gán lại vào mảng ban đầu.

```
l = l[0:2]
```

0	1
5	7

Giảm kích thước Dynamic Array

0	1	2	3	4
5	7	8	3	6

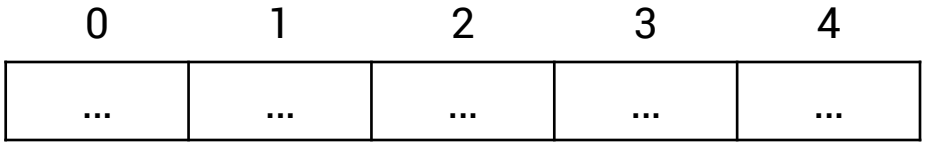
Java không có hàm resize. Do đó để giảm kích thước của ArrayList, ta sử dụng hàm subList và clear để xóa đi phần ArrayList thừa ở phía sau.

`subList(int fromIndex, int toIndex)`

```
a.subList(2, 5).clear();
```

0	1
5	7

Kiểm tra xem Dynamic Array có rỗng không



C++

python

empty()

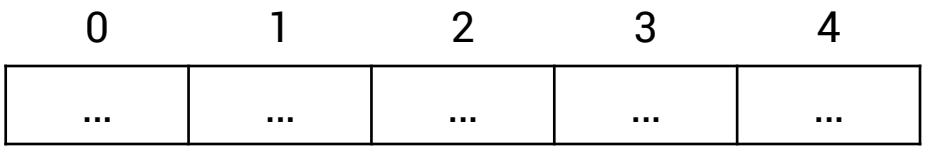
```cpp
vector<int> v;
if (v.empty() == true)
    cout << "DA is empty!";
else
    cout << "DA is not empty!";
```

Sử dụng lại hàm **len**

```python
l = []
if len(l) == 0:
    print("DA is empty!")
else:
    print("DA is not empty!")
```

DA is empty!

Kiểm tra xem Dynamic Array có rỗng không



isEmpty(): Kiểm tra ArrayList có rỗng hay không.

```
ArrayList<Integer> a = new ArrayList<Integer>();
if (a.isEmpty() == true)
    System.out.println("ArrayList is empty!");
else
    System.out.println("ArrayList is not empty!");
```

ArrayList is empty!

Duyệt xuôi trong Dynamic Array

0	1	2	3	4
5	7	8	3	6


```cpp
for(int i=0; i<v.size(); i++)
{
    cout << v[i] << ", ";
}
```

```cpp
vector<int>::iterator it;
for(it=v.begin(); it!=v.end(); it++)
{
    cout << *it << ", ";
}
```

```python
for i in range(len(l)):
    print(l[i],end=', ')
```

5, 7, 8, 3, 6,

Duyệt xuôi trong Dynamic Array

0	1	2	3	4
5	7	8	3	6


```java
for (int i = 0; i < a.size(); i++) {
    System.out.print(a.get(i) + ", ");
}
```

5, 7, 8, 3, 6,

Duyệt ngược trong Dynamic Array

0	1	2	3	4
5	7	8	3	6

C++

```cpp
for(int i=v.size()-1; i>=0; i--)
{
    cout << v[i] << ", ";
}
```

```cpp
vector<int>::reverse_iterator it;
for(it=v.rbegin(); it!=v.rend(); it++)
{
    cout << *it << ", ";
}
```

python

```python
for i in range(len(l)-1, -1, -1):
    print(l[i],end=', ')
```

6, 3, 8, 7, 5,

Duyệt ngược trong Dynamic Array

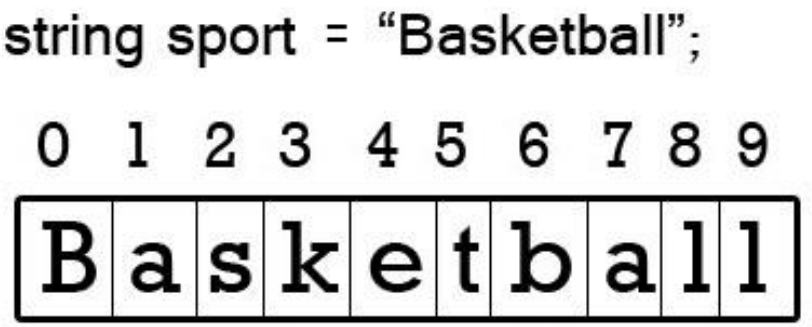
0	1	2	3	4
5	7	8	3	6


```java
for (int i = a.size() - 1; i >= 0; i--) {
    System.out.print(a.get(i) + ", ");
}
```

6, 3, 8, 7, 5,

String là gì?

String (Chuỗi) là một kiểu dữ liệu gồm nhiều ký tự (character) liên tục nhau, các ký tự ở đây rất đa dạng có thể là chữ cái, số, dấu cách, hay các ký hiệu...



C++: string

Python: variable = “ “

Java: String

Cách khai báo và sử dụng

Thư viện:

```
#include <string>
using namespace std;
```

Khai báo:

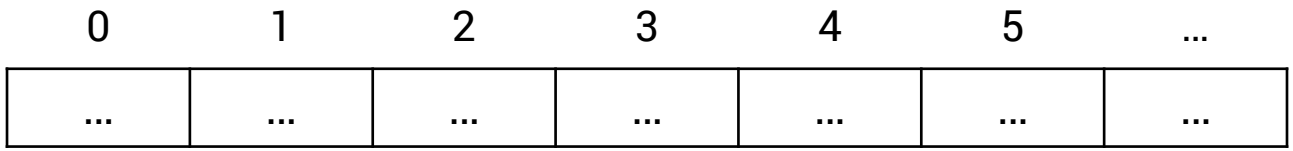
```
string variable;
```

```
string s;
```

Khai báo: Các biến trong python có thể được khởi tạo mà không cần khai báo trước kiểu dữ liệu cụ thể, tuy nhiên nên khai báo trước một chuỗi rỗng.

```
variable = ""
```

```
s = ""
```



Cách khai báo và sử dụng

Khai báo:

```
String variable = "";
```

```
String s = "";
```

Các hàm cơ bản thông dụng

1. size()/length(): Lấy kích thước của chuỗi.
2. empty(): Kiểm tra chuỗi rỗng.
3. clear(): Xóa toàn bộ chuỗi.
4. insert(pos, string): Chèn chuỗi.
5. erase(pos, len): Xóa các ký tự trong đoạn xác định.
6. find(string): Tìm kiếm chuỗi.
7. substr(pos, len): Lấy chuỗi con.
8. append(string, pos, len): Nối chuỗi.

1. len(s): Lấy kích thước của chuỗi.
2. if len(s) == 0: Sử dụng phép so sánh để kiểm tra chuỗi rỗng.
3. `s = ""` Sử dụng phép gán để xóa toàn bộ các giá trị trong chuỗi.
4. Chèn: string0 = string0[:position] + string1 + string0[position:]
5. Xóa: string = string[:start_position] + string[end_position:]
6. find(string): Tìm kiếm chuỗi.
7. Lấy chuỗi con: variable[start_pos:end_pos]
8. Ghép: Dùng phép “+” để nối chuỗi.

Các hàm cơ bản thông dụng

1. length(): Lấy kích thước của chuỗi.
2. isEmpty(): Kiểm tra chuỗi rỗng.
3. Xóa toàn bộ chuỗi: sử dụng phép gán `s = "";`
4. Sử dụng insert của **StringBuilder.**
5. Xóa: dùng hàm substring

```
s = s.substring(0, start_pos) + s.substring(end_pos);
```

6. indexOf(string): Tìm kiếm chuỗi.
7. Lấy chuỗi con: substring(start_index, end_index = length());
8. Nối chuỗi: Dùng operator +

CÁC HÀM KIỂM TRA KÝ TỰ

Bảng mã ASCII

Dec	Hex	Oct	Chr	Dec	Hex	Oct	HTML	Chr	Dec	Hex	Oct	HTML	Chr	Dec	Hex	Oct	HTML	Chr
0	0	000	NULL	32	20	040	&#032;	Space	64	40	100	&#064;	@	96	60	140	&#096;	`
1	1	001	Start of Header	33	21	041	&#033;	!	65	41	101	&#065;	A	97	61	141	&#097;	a
2	2	002	Start of Text	34	22	042	&#034;	"	66	42	102	&#066;	B	98	62	142	&#098;	b
3	3	003	End of Text	35	23	043	&#035;	#	67	43	103	&#067;	C	99	63	143	&#099;	c
4	4	004	End of Transmission	36	24	044	&#036;	$	68	44	104	&#068;	D	100	64	144	&#100;	d
5	5	005	Enquiry	37	25	045	&#037;	%	69	45	105	&#069;	E	101	65	145	&#101;	e
6	6	006	Acknowledgment	38	26	046	&#038;	&	70	46	106	&#070;	F	102	66	146	&#102;	f
7	7	007	Bell	39	27	047	&#039;	'	71	47	107	&#071;	G	103	67	147	&#103;	g
8	8	010	Backspace	40	28	050	&#040;	(	72	48	110	&#072;	H	104	68	150	&#104;	h
9	9	011	Horizontal Tab	41	29	051	&#041;	)	73	49	111	&#073;	I	105	69	151	&#105;	i
10	A	012	Line feed	42	2A	052	&#042;	*	74	4A	112	&#074;	J	106	6A	152	&#106;	j
11	B	013	Vertical Tab	43	2B	053	&#043;	+	75	4B	113	&#075;	K	107	6B	153	&#107;	k
12	C	014	Form feed	44	2C	054	&#044;	,	76	4C	114	&#076;	L	108	6C	154	&#108;	l
13	D	015	Carriage return	45	2D	055	&#045;	-	77	4D	115	&#077;	M	109	6D	155	&#109;	m
14	E	016	Shift Out	46	2E	056	&#046;	.	78	4E	116	&#078;	N	110	6E	156	&#110;	n
15	F	017	Shift In	47	2F	057	&#047;	/	79	4F	117	&#079;	O	111	6F	157	&#111;	o
16	10	020	Data Link Escape	48	30	060	&#048;	0	80	50	120	&#080;	P	112	70	160	&#112;	p
17	11	021	Device Control 1	49	31	061	&#049;	1	81	51	121	&#081;	Q	113	71	161	&#113;	q
18	12	022	Device Control 2	50	32	062	&#050;	2	82	52	122	&#082;	R	114	72	162	&#114;	r
19	13	023	Device Control 3	51	33	063	&#051;	3	83	53	123	&#083;	S	115	73	163	&#115;	s
20	14	024	Device Control 4	52	34	064	&#052;	4	84	54	124	&#084;	T	116	74	164	&#116;	t
21	15	025	Negative Ack.	53	35	065	&#053;	5	85	55	125	&#085;	U	117	75	165	&#117;	u
22	16	026	Synchronous idle	54	36	066	&#054;	6	86	56	126	&#086;	V	118	76	166	&#118;	v
23	17	027	End of Trans. Block	55	37	067	&#055;	7	87	57	127	&#087;	W	119	77	167	&#119;	w
24	18	030	Cancel	56	38	070	&#056;	8	88	58	130	&#088;	X	120	78	170	&#120;	x
25	19	031	End of Medium	57	39	071	&#057;	9	89	59	131	&#089;	Y	121	79	171	&#121;	y
26	1A	032	Substitute	58	3A	072	&#058;	:	90	5A	132	&#090;	Z	122	7A	172	&#122;	z
27	1B	033	Escape	59	3B	073	&#059;	;	91	5B	133	&#091;	[	123	7B	173	&#123;	{
28	1C	034	File Separator	60	3C	074	&#060;	<	92	5C	134	&#092;	\	124	7C	174	&#124;	|
29	1D	035	Group Separator	61	3D	075	&#061;	=	93	5D	135	&#093;	]	125	7D	175	&#125;	}
30	1E	036	Record Separator	62	3E	076	&#062;	>	94	5E	136	&#094;	^	126	7E	176	&#126;	~
31	1F	037	Unit Separator	63	3F	077	&#063;	?	95	5F	137	&#095;	_	127	7F	177	&#127;	Del

asciichars.com

Kiểm tra ký tự số

Dùng mã ASCII kiểm tra, ký tự đó thuộc vùng lưu số hay không.

```
string s("123abc");
if (s[0] >= 48 && s[0] <= 57)
    cout << "digit";
```

Trong Python sử dụng hàm ord() để lấy mã ASCII của 1 kí tự.

Để chuyển mã ASCII thành ký tự, sử dụng hàm chr().

```
s = "123abc"
if ord(s[0]) >= 48 and
ord(s[0]) <= 57:
    print("digit")
```

digit

Kiểm tra ký tự

Dùng mã ASCII kiểm tra

```
String s = "123abc";
if (s.charAt(0) >= 48 && s.charAt(0) <= 57)
    System.out.println("digit");
```

digit

Một số hàm kiểm tra ký tự

- **isalpha/isLetter:** Kiểm tra ký tự có phải chữ cái hay không.
- **isdigit/isDigit:** Kiểm tra ký tự thuộc dạng số hay không.
- **islower/isLowerCase:** Kiểm tra ký tự viết thường hay không.
- **isupper/isUpperCase:** Kiểm tra ký tự viết hoa hay không.

Lưu ý: *python không chỉ kiểm tra từng ký tự mà có thể kiểm tra toàn bộ chuỗi.*

Một số hàm kiểm tra ký tự, C++ sử dụng thư viện <ctype.h>

Kiểm tra ký tự có phải chữ cái không

isalpha

Cú pháp: int result = **isalpha**(char c)

Kết quả trả về:

- 0: ký tự đó **không** phải chữ cái.
- Khác 0: ký tự đó là chữ cái.

```
string s = "Ky Thuat Lap Trinh";
int result = isalpha(s[1]);
cout << result;
```

2

isalpha

Cú pháp: result = s.**isalpha**()

Kết quả trả về:

- False: chuỗi rỗng hoặc ký tự đó **không** phải chữ cái.
- True: ký tự đó là ký tự chữ cái.

```
s = "Ky Thuat Lap Trinh"
result = s[1].isalpha()
print(result)
```

True

Kiểm tra ký tự có phải chữ cái không

isLetter

Cú pháp: isLetter(character c). Thư viện: java.lang.Character

Kết quả trả về:

- false: ký tự đó **không** phải chữ cái.
- true: ký tự đó là chữ cái.

```
System.out.println(Character.isLetter('c'));
System.out.println(Character.isLetter('5'));
```

true
false

Các hàm kiểm tra ký tự còn lại

Cú pháp và cách sử dụng các hàm kiểm tra ký tự còn lại tương tự như hàm isalpha phía trên.

isdigit

Cú pháp: int result = **isdigit**(char c)

islower

Cú pháp: int result = **islower**(char c)

isupper

Cú pháp: int result = **isupper**(char c)

isdigit

Cú pháp: result = s.**isdigit**()

islower

Cú pháp: result = s. **islower**()

isupper

Cú pháp: result = s. **isupper**()

Các hàm kiểm tra ký tự còn lại

Cú pháp và cách sử dụng các hàm kiểm tra ký tự còn lại tương tự như hàm isLetter phía trên.

isDigit

Cú pháp: isDigit(character c)

isLowerCase

Cú pháp: isLowerCase(character c)

isUpperCase

Cú pháp: isUpperCase(character c)

CÁC HÀM CHUẨN HÓA TRONG STRING

Chuyển chuỗi thành số

atoi(char *str): Chuyển chuỗi thành số nguyên.

atof(char *str): Chuyển chuỗi thành số thực.

```
string s("12");
int number = atoi(s.c_str());
cout << number;
```

int(string): Chuyển chuỗi thành số nguyên.

float(string): Chuyển chuỗi thành số thực.

```
s = "12"
number = int(s)
print(number)
```

12

Lưu ý: trong C++ 11, bạn có thể sử dụng thêm stoi, stof

https://en.cppreference.com/w/cpp/string/basic_string/stol

Chuyển chuỗi thành số

Integer.parseInt (string, base = 10): Chuyển chuỗi thành số nguyên.

Float.parseFloat(string) Chuyển chuỗi thành số thực.

```
String s = "12";
int number = Integer.parseInt(s);
// int number = Integer.parseInt(s, 10);
System.out.println(number);
```

12

Chuyển số thành chuỗi

to_string(value)

```cpp
string s;
int number = 15789;
s = to_string(number);
cout << s;
```

str(value)

```python
number = 15789
s = str(number)
print(s)
```

15789

Chuyển số thành chuỗi

Integer.toString(value)

```
int number = 15789;
String s = Integer.toString(number);
System.out.println(s);
```

15789

In hoa và in thường ký tự - dùng hàm

tolower(char c): Chuyển ký tự thành ký tự in thường.

toupper(char c): Chuyển ký tự thành ký tự in hoa.

```
string s("algorithm");
char c = toupper(s[2]);
cout << c;
```

lower(): Chuyển ký tự thành ký tự in thường.

upper(): Chuyển ký tự thành ký tự in hoa.

Lưu ý: *trong Python hàm này có thể chuyển nguyên chuỗi thành hoa/thường.*

```
s = "algorithm"
c = s[2].upper()
print(c)
```

G

In hoa và in thường ký tự - dùng hàm

toLowerCase(): Chuyển ký tự thành ký tự in thường.

toUpperCase(): Chuyển ký tự thành ký tự in hoa.

```
String s = "algorithm";
char c = Character.toUpperCase(s.charAt(2));
System.out.print(c);
```

G

In hoa và in thường ký tự - ASCII

In hoa và in thường ký tự (**dùng mã ASCII**)

- Ký tự + 32: Chuyển ký tự thành ký tự in thường.
- Ký tự - 32: Chuyển ký tự thành ký tự in hoa.

```
string s("algorithm");
char c = s[2] - 32;
cout << s[2];
```

Lưu ý: *In hoa và in thường ký tự (**dùng mã ASCII**). String trong Python là immutable, không thể cập nhật trực tiếp vào thành phần của string để gán.*

```
s = "algorithm"
c = chr(ord(s[2]) - 32))
print(c)
```

G

In hoa và in thường ký tự - ASCII

In thường và in hoa ký tự (**dùng mã ASCII**)

- Ký tự + 32: Chuyển ký tự thành ký tự in thường.
- Ký tự - 32: Chuyển ký tự thành ký tự in hoa.

```
String s = "algorithm";
char c = s.charAt(2);
c -= 32;
System.out.print(c);
```

G

MỘT SỐ LƯU Ý KHI SỬ DỤNG STRING

Đọc từng từ và nguyên dòng

nothing is impossible


```
string s0;
cin >> s0;
cout << s0;
```

nothing

```
string s1;
getline(cin, s1);
cout << s1;
```

nothing is impossible

Đọc từng từ và nguyên dòng

nothing is impossible

Để phân tách lấy ra từng từ thì ta có thể dùng hàm split sau khi đọc.

```
line = input().split()
s0 = line[0]
print(s0)
```

nothing

```
s1 = input()
print(s1)
```

nothing is impossible

"Trong Python mặc định sẽ đọc theo từng dòng nên không có sự phân biệt giữa đọc từng từ và đọc nguyên dòng."

Đọc từng từ và nguyên dòng

nothing is impossible


```
Scanner sc = new Scanner(System.in);
String s0 = sc.next();
System.out.print(s0);
```

nothing

```
Scanner sc = new Scanner(System.in);
String s1 = sc.nextLine();
System.out.print(s1);
```

nothing is impossible

Bài toán minh họa

CODEFORCES β
Sponsored by Telegram

Fashion in Berland

Hãng thời trang Berland có quy định về quy tắc thời trang như sau:

- Các nút của chiếc áo khoác phải được cài lại, trừ 1 nút duy nhất không cài.

  ➔ Nếu đúng như vậy in ra "YES". Ngược lại "NO"

- Nếu chiếc áo khoác nào có 1 nút thì, nút đó phải được cài lại.

  ➔ Nếu đúng in ra "YES". Ngược lại in ra "NO".

Input

Dòng đầu tiên chứa số n là số lượng nút của chiếc áo khoác ($1 \le n \le 1000$).

Dòng tiếp theo lần lượt chứa n số là đại diện cho nút mở (số 0) và nút cài (số 1)

Output

In ra "YES" nếu thõa mãn yêu cầu đề bài, ngược lại in ra "NO"

Bài toán minh họa

Ví dụ:

3 1 0 1	YES

3 1 0 0	NO

Hướng dẫn giải

- **Bước 1:** Đọc toàn bộ dữ liệu đề bài vào cấu trúc dữ liệu phù hợp.

```
3
1 0 1
```

Khai báo biến n và mảng v để chứa dữ liệu đầu vào.

- **Bước 2:** Xử lý các trường hợp đặc biệt (nếu có). Trong bài toán này trường hợp đặc biệt là áo có 1 nút.
  - Nếu nút áo đó cài ➔ YES
  - Không cài ➔ NO
- **Bước 3:** Xử lý phần nội dung chính đề bài yêu cầu.
  - Duyệt vòng lặp đếm số nút = 0 (nút mở) của toàn bộ áo khoác.
  - Nếu số nút mở = 1 ➔ YES
  - Lớn hơn 1 ➔ NO

Source Code Fashion in Berland


```cpp
#include <iostream>
#include <vector>
using namespace std;
bool checkJacket(vector<int>& v, int n)
{
    if (n == 1)
        if (v[0] == 1)
            return true;
        else
            return false;
    int count = 0;
    for (int i = 0; i < n; i++)
        if (v[i] == 0)
            count++;
    if (count == 1)
        return true;
    else
        return false;
}
```

Source Code Fashion in Berland


```
20. int main()
21. {
22.     int n, value;
23.     cin >> n;
24.     vector<int> v;
25.     for (int i = 0; i < n; i++)
26.     {
27.         cin >> value;
28.         v.push_back(value);
29.     }
30.     bool result = checkJacket(v, n);
31.     if (result == true)
32.         cout << "YES";
33.     else
34.         cout << "NO";
35.     return 0;
36. }
```

Source Code Fashion in Berland

```python
def checkJacket(v, n):
    if n == 1:
        if v[0] == 1:
            return True
        else:
            return False
    count = 0
    for i in range(n):
        if v[i] == 0:
            count += 1
    if count == 1:
        return True
    else:
        return False

n = int(input())
v = list(map(int, input().split()))
if checkJacket(v, n):
    print("YES")
else:
    print("NO")
```

Source Code Fashion in Berland

```java
import java.util.ArrayList;
import java.util.Scanner;
public class Main {
    private static boolean checkJacket(ArrayList<Integer> v, int n) {
        if (n == 1) {
            if (v.get(0) == 1)
                return true;
            else
                return false;
        }
        int count = 0;
        for (int i = 0; i < n; i++) {
            if (v.get(i) == 0) {
                count++;
            }
        }
        if (count == 1)
            return true;
        else
            return false;
    }
```

Source Code Fashion in Berland

```
1.      public static void main(String[] args) {
2.          Scanner sc = new Scanner(System.in);
3.          int n = sc.nextInt();
4.          int value;
5.          ArrayList<Integer> v = new ArrayList<>();
6.          for (int i = 0; i < n; i++) {
7.              value = sc.nextInt();
8.              v.add(value);
9.          }
10.         boolean result = checkJacket(v, n);
11.         if (result == true)
12.             System.out.println("YES");
13.         else
14.             System.out.println("NO");
15.     }
16. }
```

Hỏi đáp