Arrays & Strings

Recap.

- Bits, Bytes, Encoding
- Memory management in Java
  - Stack (local vars of a function)
  - Heap (objects)

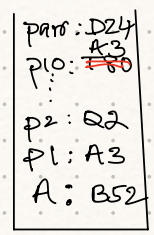
Today.

- Mem organisation for Arrays (Array indexing)
- SubArrays
- Substrings
- Sub)Sequences

} Problems.

```
int[] A = {3, 2, -5, -10}
```

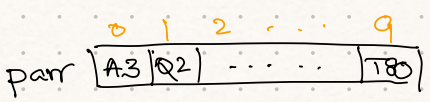
0	1	2	3
3	2	-5	-10



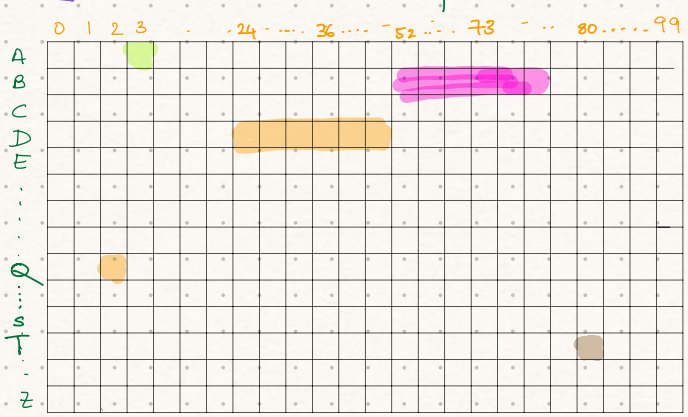
Stack

```
Point p1 = new Point(2,3);
Point p2 = "  "  "  ;
Point p10 = new Point(- -)

Point[] parr = {p1, p2, p3 . . . p10};
p10 = p1; // Assignment of Reference type
```

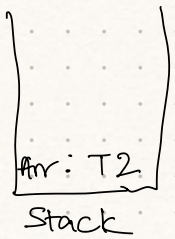


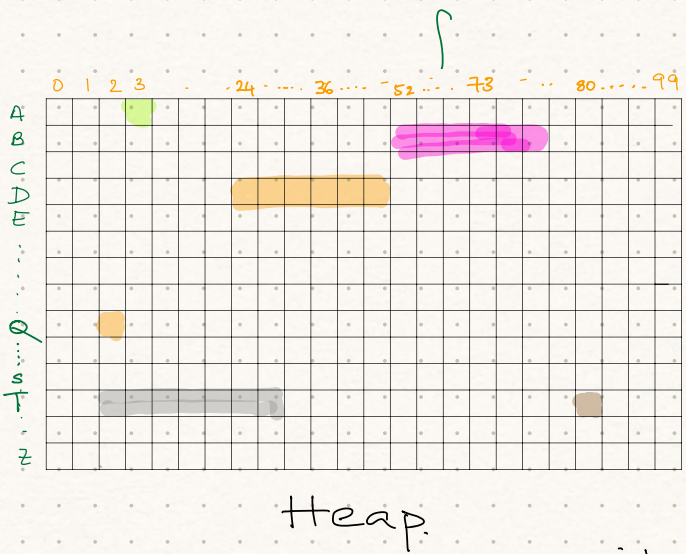
p10 will refer to the object that p1 refers to



Heap.

Array Indexing





int = 4 Bytes
32 bits

	0	1	2	3	4	5	6	7	8	9
Arr	10	-8	5	3	0	2	-4	5	6	90
	T2	T6	T10	T14	T18	T22	T26	T30	T34	T38

Addr of A[0] = T2

Addr of A[1] = T6 = T(2+4)

" " A[2] = T10 = T(2+4x2)

" " A[3] = T14 = T(2+4x3)

" " A[6] = T26 = T(2+4*6)

" " A[i] = T(2 + 4*i)

↓ addr of A[0]

A[3] = A[6]

(addr of A[3]) addr of A[0] + 4*3 = T(2+12) = T14

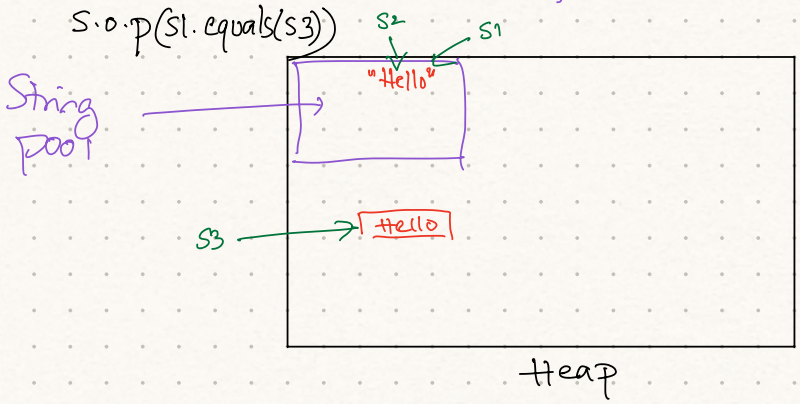
(addr of A[6]) addr of A[0] + 4*6 = T(2+4*6) = T(26)

Addr of element at index i = Addr of element at index 0 + (Size of element) * i

A[1000]

Strings

```
String s1 = "Hello";
String s2 = "Hello";
s.o.p (s1 == s2);          // true (Do s1 & s2 refer to same obj?)
String s3 = new String ("Hello");
s.o.p (s1 == s3)           // false. (Do s1 & s3 refer to same obj)
s.o.p(s1.equals(s3))
```



Substrings

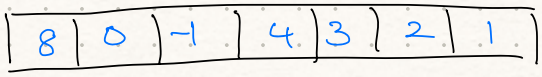
S = "a b c d e f g h i"

Substring of S = any continuous sequence of characters of S

Find all substrings of "abcd"

Start with a	a , ab, abc, abcd	10 substrings of abcd.
Start with b	b, bc, bcd.	
Start with c	c , cd	
Start with d	d	

Subarrays



↓

continuous elements in the array

Subsequence

- resulting sequence obtained by deleting zero or more elements of a given sequence.

S = a b c d e f g h i

a d f h i ✓

e f g h i ✓

b e h ✓

b g e ✗

a i f ✗

10 8 , 3 , 5

3 , 10 ✗

Given string S S = "_ _ _ _ _"

s1 = a substring of S

s2 = a subsequence of S

1. Is s1 a subsequence of S ? Yes (always)

2. Is s2 a substring of S ? May or may not be

S = a b c d

s2 = b d ---> not a substring

s2 = c d ---> is a substring

Ex: Write a program to print all substrings of a given string S.

S = "0 1 2 3 4 " a b c d e (use substring method of String class)

S.substring(1, 4) // bcd

S.substring(2, 5) // cde

S = "a b c d"
0 1 2 3

i	j	Output	
0	1	a	s.substring(0, 1)
0	2	a b	s.substring(0, 2)
0	3	a b c	s.substring(0, 3)
0	4	a b c d	s.substring(0, 4)
1	2	b	s.substring(1, 2)
1	3	b c	s.substring(1, 3)
1	4	b c d	s.substring(1, 4)
2	3	c	
2	4	c d	
3	4	d	

```
for (int i = 0; i < s.length(); i++) {
    for (int j = i+1; j <= s.length(); j++)
        s.o.p (s.substring (i, j));
}
```

Write programs:

S = "abcd"

1. Count the number of substrings of S // Output: 10

2. Count the number of substrings that start & end with same char

S = "abaeba"

3. Count the number of substrings which are palindromes.

S = "ababab"

```
for(int i = 0; i < s.length(); i++) {
    for(int j = i+1; j <= s.length(); j++)
        s.o.p(s.substring(i, j));
}
```

S = "abcd"
0 1 2 3

i	i < s.length() (4)	Body				i++	Output:
0	0 < 4 ✓	j	j <= s.length() (4)	Body	j++		a
		1	1 <= 4 ✓	substring(0,1)	2		ab
		2	2 <= 4 ✓	substring(0,2)	3	1	abc
		3	3 <= 4 ✓	substr(0,3)	4		abcd
		4	4 <= 4 ✓	substr(0,4)	5		
		5	5 <= 4 ✗				
1	1 < 4 ✓	j	j <= s.length()	Body	j++		b
		2	2 <= 4 ✓	substr(1,2)	3		bc
		3	3 <= 4 ✓	substr(1,3)	4		bcd
		4	4 <= 4 ✓	substr(1,4)	5		