Public-Service Announcement I

an engineer, designer, or entrepreneur? Come check evelopers of Berkeley. We create end-to-end solu- diverse breath of skills. Our members hone their tend, backend, design, and engineering while devel- erstanding of topics such as software architecture, perating systems, and networking. We build apps wide range of industries while leveraging the latest in IOT, AI, fintech, ML, AR, and more. Join us for e tonight, Wednesday 9/6. Our members will show and what it took to make them. Free food, fidget wag will also be there. s://www.mobiledevsberkeley.org/ for further infor-

Public-Service Announcement II

Political Computer Science @ Berkeley! PCS har- nts' intellectual capabilities and potential in CS and affairs to address current issues within the United and political systems.

dy have an Amazon Alexa Skill called Political Pun- bout to be published; it is able to give responses to ut politics/government.

he many other projects that we have in the works o gerrymandering, campaign finance, Alexa, and more), site: pcsberkeley.wixsite.com/pcsberkeley

w accepting applications for the 2017-2018 school takes 5 minutes to complete)! Apply on our website!"

Recreation

n of the coefficients of

$$(1-3x+3x^2)^{743}(1+3x-3x^2)^{744}$$

and collecting terms?

CS61B Lecture #6: Arrays

structured container whose components are

fixed integer.

e of **length** simple containers of the same type, num- m 0.

eld usually implicit in diagrams.)

nonymous, like other structured containers.

red to with pointers.

inted to by `A`,

`A.length`

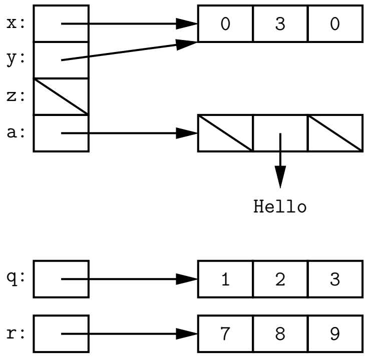
d component i is `A[`i`]` (i is the *index*)

t feature: index can be *any integer expression*.

A Few Samples

Java **Results**

```
;
;
g[3];
';
{ 1, 2, 3 };
for declarations:
 8, 9 };
```



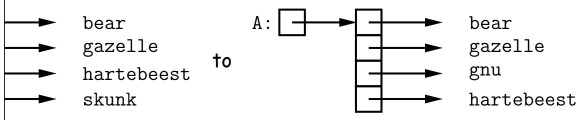
Example: Accumulate Values

up the elements of array A.

```
n(int[] A) {
                                         // New (1.5) syntax
= 0; i < A.length; i += 1)               for (int x : A)
                                            N += x;


d-core: could have written

 i<A.length; N += A[i], i += 1)
just ;

don't: it's obscure.
```

Example: Insert into an Array

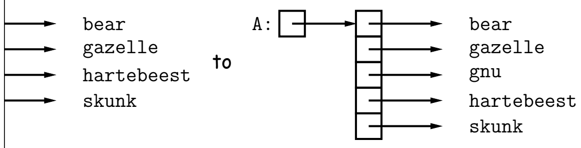
t a call like `insert(A, 2, "gnu")` to convert (destruc-



```
location K in ARR, moving items K, K+1, ... to locations
..  The last item in ARR is lost. */
rt (String[] arr, int k, String x) {
rr.length-1; i > k; i -= 1) // Why backwards?
[i-1];
 to this loop:
rraycopy(arr, k,  arr, k+1,  arr.length-k-1);*/
          from      to         # to copy
```

Growing an Array

ose that we want to change the description above, so t2 (A, 2, "gnu") does *not* shove "skunk" off the end, ws" the array.



```
, r, where r.length = ARR.length+1; r[0..K-1]
ARR[0..K-1], r[k] = x, r[K+1..] same as ARR[K..]. */
insert2(String[] arr, int k, String x) {
t = new String[arr.length + 1];
 0, result, 0, k);
 k, result, k+1, arr.length-k);
```

l a different return type from insert2??

Example: Merging Program

n two sorted arrays of ints, A and B, produce their l array containing all from A and B.

ler to solve this recursively, it is useful to *generalize* ction to allow merging *portions* of the arrays.

```
nd B are sorted, returns their merge. */
t[] merge(int[] A, int[] B) {
A, 0, B, 0);


 A[L0..] and B[L1..] assuming A and B sorted. */
ge(int[] A, int L0, int[] B, int L1) {
gth - L0 + B.length - L1; int[] C = new int[N];
ength) arraycopy(B, L1, C, 0, N);
= B.length) arraycopy(A, L0, C, 0, N);
] <= B[L1]) {
0]; arraycopy(merge(A, L0+1, B, L1), 0, C, 1, N-1);

1]; arraycopy(merge(A, L0, B, L1+1), 0, C, 1, N-1);
```

What is wrong with this implementation?

A Tail-Recursive Solution

```
t[] merge(int[] A, int[] B) {
A, 0, B, 0, new int[A.length+B.length], 0);


] and B[L1..] into C[K..], assuming A and B sorted. */
ge(int[] A, int L0, int[] B, int L1, int[] C, int k){
ength) /* ? */
= B.length) /* ? */
] <= B[L1]) {
0];


1];
```

(Aside) Java Shortcut

Can write just 'arraycopy' by including at the top of the

```
ic java.lang.System.arraycopy;
```

define the simple name arraycopy to be the equivalent g.System.arraycopy in the current source file."

ame for out so that you can write

```
(...);


println(...);
```

laration like

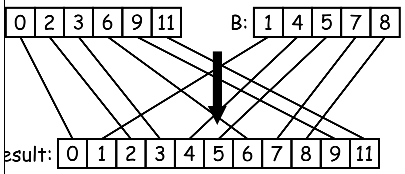
```
ic java.lang.Math.*;
```

all the (public) static definitions in java.lang.Math and available in this source file by their simple names (the he last dot)."

unctions like sin, sqrt, etc.

Example: Merging

n two sorted arrays of ints, A and B, produce their l array containing all items from A and B.


0 | 2 | 3 | 6 | 9 | 11

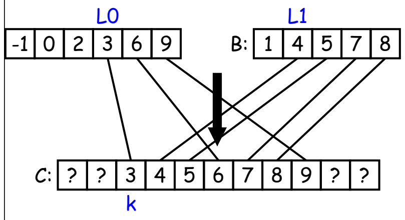
B: 1 | 4 | 5 | 7 | 8

esult: 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 11

A Tail-Recursive Strategy

```
t[] merge(int[] A, int[] B) {
A, 0, B, 0, new int[A.length+B.length], 0);


] and B[L1..] into C[K..], assuming A and B sorted. */
ge(int[] A, int L0, int[] B, int L1, int[] C, int k){
```

d merges *part* of A with part of B into part of C. For er a possible call merge(A, 3, B, 1, C, 2)



-1 | 0 | 2 | 3 | 6 | 9

B: 1 | 4 | 5 | 7 | 8

C: ? | ? | 3 | 4 | 5 | 6 | 7 | 8 | 9 | ? | ?

A Tail-Recursive Solution

```
t[] merge(int[] A, int[] B) {
A, 0, B, 0, new int[A.length+B.length], 0);


] and B[L1..] into C[K..], assuming A and B sorted. */
ge(int[] A, int L0, int[] B, int L1, int[] C, int k){
ength) /* ? */
= B.length) /* ? */
] <= B[L1]) {
0];


1];
```

A Tail-Recursive Solution

```
t[] merge(int[] A, int[] B) {
A, 0, B, 0, new int[A.length+B.length], 0);


] and B[L1..] into C[K..], assuming A and B sorted. */
ge(int[] A, int L0, int[] B, int L1, int[] C, int k){
ength) arraycopy(B, L1, C, k, B.length-L1);
= B.length) arraycopy(A, L0, C, k, A.length-L0);
] <= B[L1]) {
0];


1];
```

A Tail-Recursive Solution

```
t[] merge(int[] A, int[] B) {
A, 0, B, 0, new int[A.length+B.length], 0);


] and B[L1..] into C[K..], assuming A and B sorted. */
ge(int[] A, int L0, int[] B, int L1, int[] C, int k){
ength) arraycopy(B, L1, C, k, B.length-L1);
= B.length) arraycopy(A, L0, C, k, A.length-L0);
] <= B[L1]) {
0];
0+1, B, L1, C, k+1);

1];
0, B, L1+1, C, k+1);
```

Iterative Solution

lon't use either of the previous approaches in languages
Array manipulation is most often iterative:

```
int[] merge(int[] A, int[] B) {
ew int[A.length + B.length];
```

Iterative Solution II

```
t[] merge(int[] A, int[] B) {
 int[A.length + B.length];


0; k < C.length; k += 1) {
 A.length) {
= B[L1]; L1 += 1;
 (L1 >= B.length) {
= A[L0]; L0 += 1;
 (A[L0] <= B[L1]) {
= A[L0]; L0 += 1;

= B[L1]; L1 += 1;
```

lternative Solution: Removing k

ant of the loop is that k=L0+L1.

```
t[] merge(int[] A, int[] B) {
 int[A.length + B.length];


1 < C.length) {
 A.length) {
+ L1] = B[L1]; L1 += 1;
 (L1 >= B.length) {
+ L1] = A[L0]; L0 += 1;
 (A[L0] <= B[L1]) {
+ L1] = A[L0]; L0 += 1;

+ L1] = B[L1]; L1 += 1;
```

Multidimensional Arrays

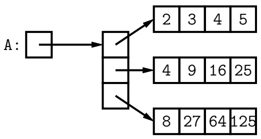
- or higher-dimensional layouts, such as

$$A = \begin{array}{|r|r|r|r|}\hline 2 & 3 & 4 & 5 \\ \hline 4 & 9 & 16 & 25 \\ \hline 8 & 27 & 64 & 125 \\ \hline \end{array} \quad ?$$

Multidimensional Arrays in Java

primitive in Java, but we can build them as arrays of

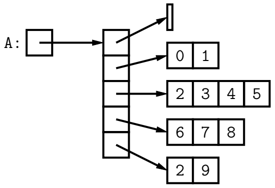
```
ew int[3][];
t[] {2, 3, 4, 5};
t[] {4, 9, 16, 25};
t[] {8, 27, 64, 125};


[] { {2, 3, 4, 5},
     {4, 9, 16, 25},
     { 8, 27, 64, 125} };

 {2, 3, 4, 5},
 {4, 9, 16, 25},
 {8, 27, 64, 125} };

ew A[3][4];
0; i < 3; i += 1)
j = 0; j < 4; j += 1)
j] = (int) Math.pow(j + 2, i + 1);
```



Exotic Multidimensional Arrays

element of an array is independent, there is no single
neral:

```
= new int[5][];
 int[] {};
 int[] {0, 1};
 int[] {2, 3, 4, 5};
 int[] {6, 7, 8};
 int[] {9};
```



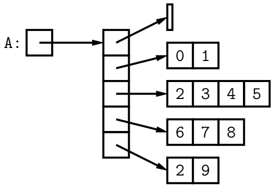
his print?

```
RO = new int[3][];
ZERO[1] = ZERO[2] =
t[] {0, 0, 0};
 = 1;
.println(ZERO[2][1]);
```

Exotic Multidimensional Arrays

element of an array is independent, there is no single
neral:

```
= new int[5][];
 int[] {};
 int[] {0, 1};
 int[] {2, 3, 4, 5};
 int[] {6, 7, 8};
 int[] {9};
```



his print?

```
RO = new int[3][];
ZERO[1] = ZERO[2] =
t[] {0, 0, 0};
 = 1;
.println(ZERO[2][1]);
```