Today's plan

- Remainder:
  - Find your mid-term project partner and register your team on Piazza.
  - Homework 4 dues tomorrow (March 2nd).
- Review
- Ex 6-1

Ex 5-3 - random integers

Which of the following generate a random integer between -5 and 8 inclusively?

A `rand() % 8 - 5`

B `rand() % 13 - 5`

C `rand() % 14 - 5`

D `rand() / RAND_MAX * 13 - 5`

E `(double)rand() / RAND_MAX * 13 - 5`

Ex 5-3 - two's complement

Given a 8 bits `unsigned char value`, which of the following compute it's two's complement?

A `~value | 1;`

B `!value | 1;`

C `~value + 1;`

D `!value + 1;`

E `256 - value;`

Ex 5-3 - little and big endian

If we use the below code to write an integer array to a binary file on a little endian machine (e.g. x86 CPU)

```
int data[5] = {1, 2, 3, 4, 5};
FILE *fp = fopen("wb");
fwrite(data, sizeof(int), 5, fp);
fclose();
```

Can we use the below code to read the array from the binary file on a big endian machine (e.g. Arm64 CPU)?

```
int data[5] = {0};
FILE *fp = fopen("rb");
fread(data, sizeof(int), 5, fp);
fclose();
```

A Yes

B No

Reference: https://en.wikipedia.org/wiki/Comparison_of_

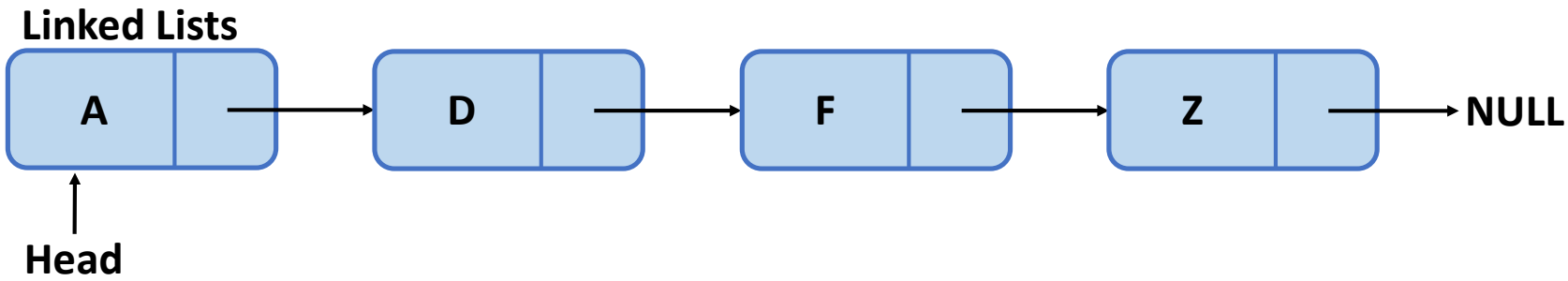
Ex 5-3 - little and big endian

What does the below code do?

```
int data[5] = {1, 2, 3, 4, 5};
char* ptr = NULL;
for (int i = 0; i < 5; ++i) {
    ptr = (char*)&data[i];
    for (int j = 0; j < sizeof(int) / 2; ++j) {
        char temp = ptr[j];
        ptr[j] = ptr[sizeof(int) - j - 1];
        ptr[sizeof(int) - j - 1] = temp;
    }
}
```

Linked lists

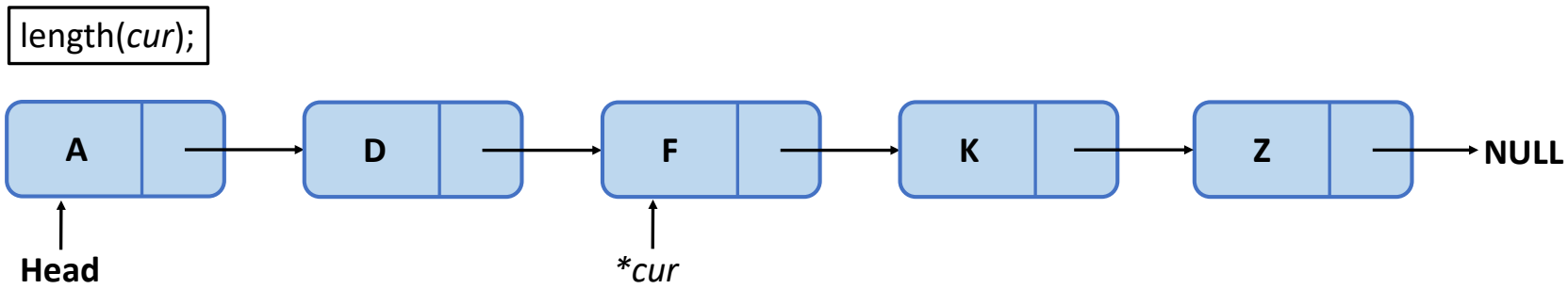
Describe the linked list structure using a diagram.



How is the linked list's head different from a node?

Linked lists

How do you calculate the `length` of a linked list?

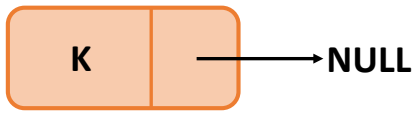


Linked lists

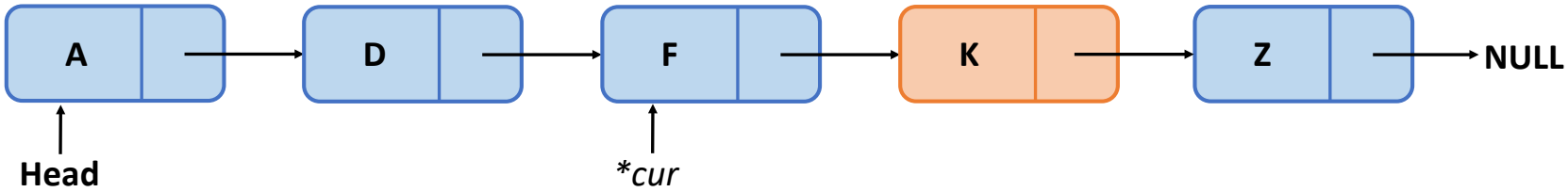
How do you implement the `add_after` function for a linked list?

add_after(*cur*, 'K');

1) Create a new node



2) Add the new node after **cur*



Linked lists

Which of the following about arrays and linked lists is true?

A Inserting new element into a linked list is faster than into an array.

B We can dynamically change the size of an array but not a linked list.

C Accessing the i'th element in a linked list is faster than accessing that in an array.

D Removing a node in a linked list is easier than removing an element in an array.

E Linked list requires more space than arrays.

Linked lists

Consider the following program.

```
1  #include <stdio.h>
2  #include <stdlib.h>
3  typedef struct node_ {
4      char data;
5      struct node_ *next;
6  } Node;
7  int main(void) {
8      Node *a = malloc(sizeof(Node)),
9      *b = malloc(sizeof(Node)),
10     *n;
11     a->data = 'A';
12     b->data = 'B';
13     a->next = b;
14     b->next = a;
15     for (n = a; n != NULL; n = n->next) {
16         printf("%c ", n->data);
17     }
18     printf("\n");
19     return 0;
20 }
```

What output is printed?

A No output is printed.

B A

C A B

D B A

E None of the above.

Class exercises

Ex 6-1