CS61C Fall 2025

C, Floating Point

Discussion 3

1 Floating Point

1.1 Convert the following single-precision floating point numbers from hexadecimal to decimal or from decimal to hexadecimal using the IEEE 754 Floating Point Standard. You may leave your answer as an expression.

a) `8.25`

b) `39.5625`

c) `0x00000F00`

d) `0`

e) `0xFF94BEEF`

f) ∞

g) `1/3`

2 More Floating Point

As we saw above, not every number can be represented perfectly using floating point. For this question, we will only look at positive numbers.

2.1 What is the next smallest number larger than 2 that can be represented completely?

2.2 What is the next smallest number larger than 4 that can be represented completely?

2.3 What is the largest odd number that we can represent? Hint: at what power can we only represent even numbers?

3 C Generics

3.1 **True** or **False**: In C, it is possible to directly dereference a `void *` pointer, e.g.

```
... = *ptr;
```

3.2 Generic functions (i.e., generics) in C use `void *` pointers to operate on memory without the restriction of types. Generic pointers do not support dereferencing, as the number of bytes to access from memory is not known at compile-time. They instead use byte handling functions such as `memcpy` and `memmove`.

Implement `rotate`, which will prompt the following program to generate the provided output.

```
int main(int argc, char *argv[]) {
  int array[10] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10};
  print_int_array(array, 10);
  rotate(array, array + 5, array + 10);
  print_int_array(array, 10);
  rotate(array, array + 1, array + 10);
  print_int_array(array, 10);
  rotate(array + 4, array + 5, array + 6);
  print_int_array(array, 10);
  return 0;
}
```

Output:

```
$ ./rotate
Array: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
Array: [6, 7, 8, 9, 10, 1, 2, 3, 4, 5]
Array: [7, 8, 9, 10, 1, 2, 3, 4, 5, 6]
Array: [7, 8, 9, 10, 2, 1, 3, 4, 5, 6]
```

Your Solution:

```
void rotate(void *front, void *separator, void *end) {




}
```