1.3

```
// Assume n holds the length of arr
double fast_product(double *arr, int n) {
    double product = 1;
    #pragma omp parallel for
    for (int i = 0; i < n; i++) {
        product *= arr[i];
    }
    return product;
}
```

(a) What is wrong with this code?

The code has the shared variable product.

(b) Fix the code using `#pragma omp critical`

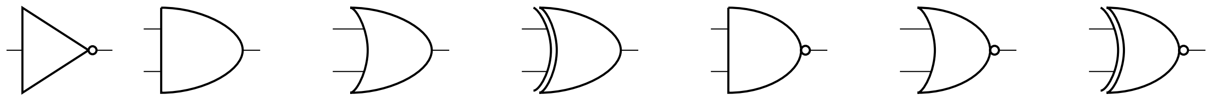
```
double fast_product(double *arr, int n) {
    double product = 1;
    #pragma omp parallel for
    for (int i = 0; i < n; i++) {
        #pragma omp critical
        product *= arr[i];
    }
    return product;
}
```

(c) Fix the code using `#pragma omp reduction(operation: var)`.

```
double fast_product(double *arr, int n) {
    double product = 1;
    #pragma omp parallel for reduction(*: product)
    for (int i = 0; i < n; i++) {
        product *= arr[i];
    }
    return product;
}
```

2 Logic Gates

2.1 Label the following logic gates:



NOT, AND, OR, XOR, NAND, NOR, XNOR

2.2 Convert the following to boolean expressions:

(a) NAND