```
parse_line (generic function with 1 method)
```

```julia
function parse_line(line::String)
    words = split(line)

    return (words[1], parse(Int32, words[2]))
end
```

Problem 1

```
calculate_position (generic function with 1 method)
```

```julia
function calculate_position(io::IO)
    horizontal, depth = 0, 0
    for line in eachline(io)
        (direction, val) = parse_line(line)
        if (direction == "forward")
            horizontal += val
        else
            depth += ((direction == "down") ? 1 : -1) * val
        end
    end

    return (horizontal, depth)
end

```

```
2322630
```

```julia
open("./Day2/prob_1_input.txt") do io
    (horizontal, depth) = calculate_position(io)

    horizontal * depth
end
```

Problem 2

```
calculate_aim_position (generic function with 1 method)
```

```julia
function calculate_aim_position(io::IO)
    horizontal, depth, aim = 0, 0, 0
    for line in eachline(io)
        (direction, val) = parse_line(line)
        if (direction == "forward")
            horizontal += val
            depth += aim * val
        else
            aim += ((direction == "down") ? 1 : -1) * val
        end
    end

    return (horizontal, depth)
end
```

```
2105273490
```

```julia
open("./Day2/prob_1_input.txt") do io
    (horizontal, depth) = calculate_aim_position(io)

    horizontal * depth
end
```