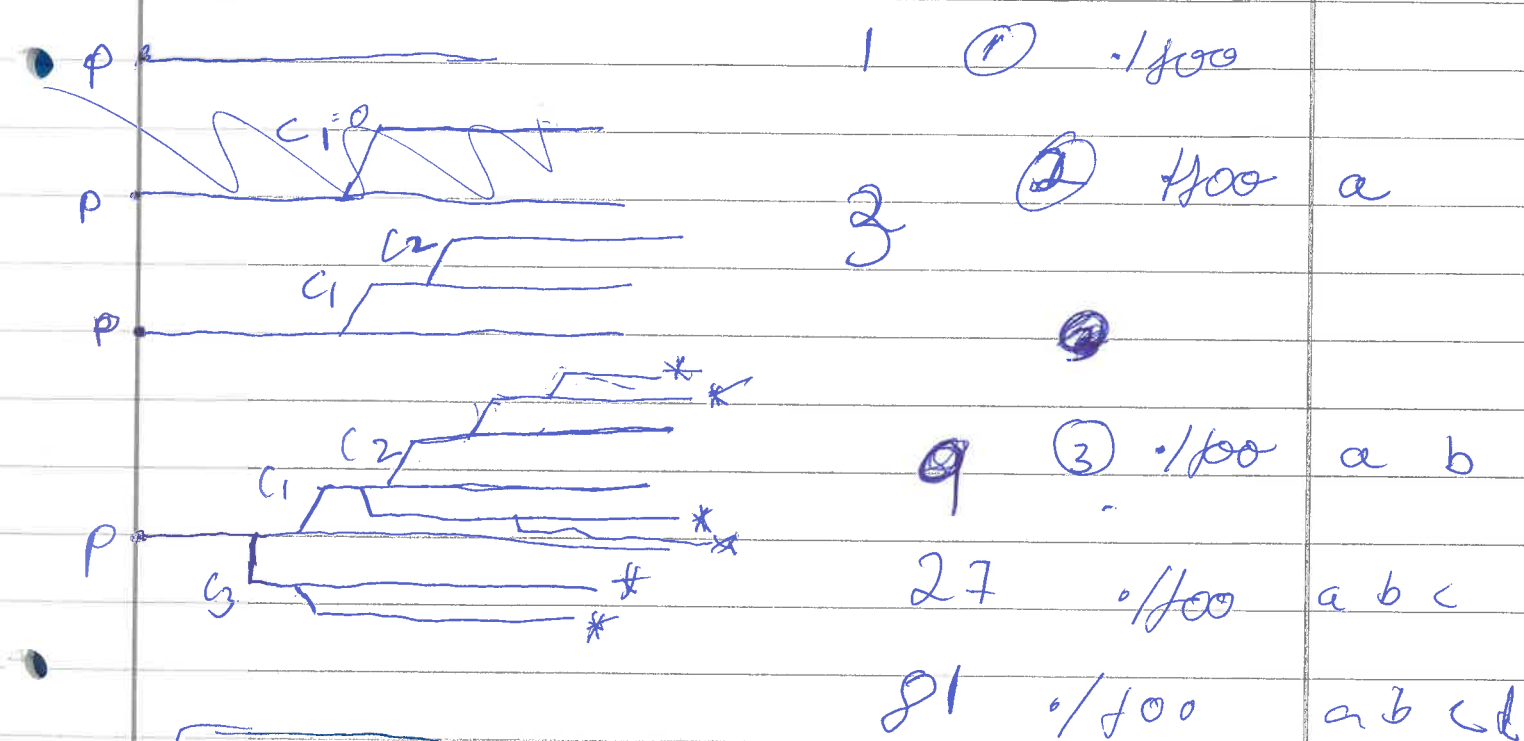


Q1

```

for(; argc > 1; argc--) {
    if (0 == fork()) {
        (void) fork();
    }
}
return 0
    
```

- ①  $\cdot / \text{foo}$   $\text{argc}$  is 1 never enters for loop
- ②  $\cdot / \text{foo } a$   $\text{argc} = 2$   $\begin{cases} \text{argv}[0] = 0 \\ \text{argv}[1] = a \end{cases}$  so one iteration  
2 args
- ③  $\cdot / \text{foo } a \ b$   $\text{argc} = 3$   $\begin{cases} \text{argv}[0] = 0 \\ \text{argv}[1] = a \\ \text{argv}[2] = b \end{cases}$   $\begin{matrix} 3-1=2 \\ 2-1=1 \end{matrix}$  = ① ② iter.



So # of child processes is  $81 - 1 = 80$

11.2

4 \* 2 \* x + 7

- a)
- $$\begin{aligned}
 &\langle \text{digit} \rangle * [(\langle \text{digit} \rangle * \langle \text{var} \rangle) + \langle \text{digit} \rangle] \\
 &\langle \text{constant} \rangle * [(\langle \text{digit} \rangle * \langle \text{var} \rangle) + \langle \text{digit} \rangle] \\
 &\langle \text{factor} \rangle * [(\langle \text{digit} \rangle * \langle \text{var} \rangle) + \langle \text{digit} \rangle] \\
 &\langle \text{term} \rangle * [(\langle \text{digit} \rangle * \langle \text{var} \rangle) + \langle \text{digit} \rangle] \\
 &\langle \text{term} \rangle * [(\langle \text{const} \rangle * \langle \text{factor} \rangle) + \langle \text{constant} \rangle] \\
 &\langle \text{term} \rangle * [(\langle \text{factor} \rangle * \langle \text{factor} \rangle) + \langle \text{factor} \rangle] \\
 &\langle \text{term} \rangle * [(\langle \text{term} \rangle * \langle \text{factor} \rangle) + \langle \text{term} \rangle] \\
 &\langle \text{term} \rangle * [\langle \text{expression} \rangle + \langle \text{term} \rangle] \\
 &\langle \text{term} \rangle * \langle \text{expression} \rangle \\
 &\langle \text{term} \rangle * \langle \text{factor} \rangle \\
 &\langle \text{term} \rangle \\
 &\langle \text{expression} \rangle =
 \end{aligned}$$

- b) Yes, the associativity of the multiplication so basically adding the <sup>extra</sup> brackets eg.  $(2 * x)$ , to make it clear that we are multiplying first instead of adding then multiplying.