GitHub Repository: https://github.com/edwardinio18/LFTC

L1a

p1.txt

```
// Min of 3 numbers

cvnou €a: intreg;
cvnou €b: intreg;
cvnou €c: intreg;

citeste(€a);
citeste(€b);
citeste(€c);

€smallest_int: intreg;
€smallest_int estiegal €a;

daca (€smallest_int maimare €b) atunci {
    €smallest_int estiegal €b;
}

daca (€smallest_int maimare €c) atunci {
    €smallest_int estiegal €c;
}

scrie(€smallest_int);
```

p1err.txt

```
cvnou a: intreg; // variable must begin with €
cvnou €b: intreg;
cvnou €c: intreg;

citeste(€a);
citeste(€b);
citeste(€c);

€smallest_int: intreg;
€smallest_int = €a;

dacaa (€smallest_int maimaree €b) atunci { // daca spelled wrong; maimare spelled wrong
    €smallest_int = €b;
}

daca (€smallest_int maimare €c) atunci {
    €smallest_int = €c;
}

scrie(€smallest_int);
```

p2.txt

```
// Verify if a number is prime, compute gcd of 2 numbers, compute the solutions for a 2nd order equation

// Verify if a number is prime
cvnou €x: intreg;
citeste(€x);
cvnou €i: intreg;
€i estiegal 2;
cvnou €prime: intreg;
€prime estiegal 1;

cattimp (€i oriori €i maimicegal €x sisi €prime verificaegal 1) fa {
    daca (€x lasuta €i verificaegal 0) atunci {
        €prime estiegal 0;
    }
    €i estiegal €i adunate 1;
}

daca (€prime verificaegal 1) atunci {
    scrie("Yes, it is prime!");
} altfel {
    scrie("No, it is not prime.");
}

// Compute gcd of 2 numbers
cvnou €a: intreg;
cvnou €b: intreg;

scrie("a=");
citeste(€a);

scrie("b=");
citeste(€b);

cvnou €remainder: intreg;

cattimp (€b verificanuegal 0) fa {
    €remainder estiegal €a lasuta €b;
    €a estiegal €b;
    €b estiegal €remainder;
}

scrie("GCD is ");
scrie(€a);

// Compute the solutions for a 2nd order equation
cvnou €a: intreg;
cvnou €b: intreg;
cvnou €c: intreg;

scrie("The ecuation: a*x^2 + b*x + c = 0\n");

scrie("a=");
citeste(€a);

scrie("b=");
citeste(€b);

scrie("c=");
citeste(€c);

cvnou €d: intreg;
€d estiegal €b oriori €b stergete 4 oriori €a oriori €c;

cvnou €sol1: real;
cvnou €sol2: real;

daca €d maimic 0 atunci {
    scrie("No real solutions.");
}

daca €d verificaegal 0 atunci {
    €sol1 estiegal (stergete€b) orioriinvers (2 oriori €a);
    scrie("Unique solution ");
    scrie(€sol1);
}

€sol1 estiegal ((stergete€b) stergete (€d)) orioriinvers (2 oriori €a);
€sol2 estiegal ((stergete€b) adunate (€d)) orioriinvers (2 oriori €a);

scrie("2 solutions ");
scrie(€sol1);
scrie(" and ");
scrie(€sol2);
```

p3.txt

```
// Compute the sum of n numbers, compute the max/min of n numbers

// Compute the sum of n numbers
cvnou €n: intreg;
cvnou €sum: intreg;
cvnou €i: intreg;

citeste(€n);
€sum estiegal 0;
€i estiegal 1;

cattimp (€i maimicegal €n) fa {
    €sum estiegal €sum adunate €cvnou;
    €i estiegal €i adunate 1;
}

scrie("Sum: ");
scrie(€sum);

// Compute the max/min of n numbers
cvnou €numbers: sir(10) de intreg;
cvnou €max_num: intreg;
cvnou €min_num: intreg;

citeste(€numbers);
€max_num estiegal stergete 999999;
€min_num estiegal 999999;

pentrufiecare (€num in €numbers) fa {
    daca (€num maimare €max_num) atunci {
        €max_num estiegal €num;
    }
    daca (€num maimic €min_num) atunci {
        €min_num estiegal €num;
    }
}

scrie("Max: ");
scrie(€max_num);
scrie("Min: ");
scrie(€min_num);
```