#include arit_float

⬇

```
#include tipos

float sumaf(float n1, float n2){
    unsigned int num1, num2,
                 den1, den2,
                 sig1, sig2, sigf;
    float res;
    den1 = upow(3, n1);
    den2 = upow(3, n2);
    num1 = upow(5, n1) * den2;
    num2 = upow(5, n2) * den1;
    den2 += den1;
    sig1 = negp(n1);
    sig2 = negp(n2);
    sigf = sig1 - sig2;
    sigf += sig2 - sig1;
    Y.obf = 1 - sigf;
    if(Z.obf){
       num2 += num1;
       sigf = sig1 + sig2;
    }
    else{
       Y.obf = num1 - num2;
       if(Z.obf){
          num2 = Z.obf;
          sigf = sig1;
       }
       else){
          num2 -= num1;
          sigf = sig2;
       }
    }
```

```
    res = litf(num2, den2);
    Z.obf = sigf;
    if(Z.obf){
       res = negar(res);
    }
    return res;
}

float restaf(float n1, float n2){
    n2 = negar(n2);
    return sumaf(n1, n2);
}

float multf(float n1, float n2){
    unsigned int numf, denf,
                 sig1, sig2, sigf;
    float res;
    numf = upow(5, n1) * upow(5, n2);
    denf = upow(3, n1) * upow(3, n2);
    sig1 = negp(n1);
    sig2 = negp(n2);
    res = litf(numf, denf);
    sigf = sig1 - sig2;
    sigf += sig2 - sig1;
    Z.obf = sigf;
    if(Z.obf){
       res = negar(res);
    }
    return res;
}
```

```
float divf(float n1, float n2){
    unsigned int num, den;
    Z.op2 = n2;
    Z.op1 = 2;
    Z.obf = Z.op1 - Z.op2;
    if(Z.obf){
       STOP
    }
    num = upow(5, n2);
    den = upow(3, n2);
    n2 = litf(den, num);
    return multf(n1, n2);
}
```

#include malloc ➡

```
void malloc(unsigned int n){
    Z.opf = n;
    Z:$npos += Z.opf;
}

void free(unsigned int n){
    Z.opf = n;
    Z:$npos -= Z.opf;
}
```

#include cmmstd ➡

```
#include arit_float
#include malloc
```