Especificación formal das funcións

A especificación farase en función da lóxica de Hoare, usando o triplete {P} S {Q}.

Funcións:

- bignum str2bignum (char *str)
- bignum add (bignum a, bignum b)
- bignum sub (bignum a, bignum b)
- bignum mult (bignum a, bignum b)
- bignum remainder1 (bignum a, bignum n)
- bignum fact (bignum n)
- bignum multmod (bignum a, bignum b, bignum n)
- int comparar (bignum a, bignum b)
- void imprimir(bignum resultado)

1. bignum str2bignum (char *str);

bignum totmult;

char *str;

{P} ≡ {a.val != NULL AND b.val != NULL AND b.tam>0 AND a.tam>0 AND 0<=a.sign<=1 AND 0<=b.sign<=1 AND (∀i: 0<=i<a.tam: 0<=a.val[i]<=9) AND (∀i: 0<=i<b.tam: 0<=b.val[i]<=9)}

S ≡ mult

{Q} ≡ {totmult.tam>0 AND 0<=totmult.sign<=1 AND totmult.val!=NULL AND (∀i: 0<=i<totmult.tam: 0<=totmult.val[i]=9) AND totmult==a*b}

2. bignum add (bignum a, bignum b);

bignum a, b, totsum;

{P} ≡ {a.val != NULL AND b.val != NULL AND b.tam>0 AND a.tam>0 AND 0<=a.sign<=1 AND 0<=b.sign<=1 AND (∀i: 0<=i<a.tam: 0<=a.val[i]<=9) AND (∀i: 0<=i<b.tam: 0<=b.val[i]<=9)}

S ≡ add

{Q} ≡ {totsum.tam>0 AND 0<=totsum.sign<=1 AND totsum.val != NULL AND (∀i: 0<=i<num.tam: 0<=num.val[i]=9) AND totsum==a+b}

3. bignum sub (bignum a, bignum b);

bignum a, b, totrest;

{P} ≡ {a.val != NULL AND b.val != NULL AND b.tam>0 AND a.tam>0 AND 0<=a.sign<=1 AND 0<=b.sign<=1 AND (∀i: 0<=i<a.tam: 0<=a.val[i]<=9) AND (∀i: 0<=i<b.tam: 0<=b.val[i]<=9)}

S ≡ sub

{Q} ≡ {totrest.tam>0 AND 0<=totrest.sign<=1 AND totrest.val != NULL AND (∀i: 0<=i<totrest.tam: 0<=totrest.val[i]=9) AND totrest==a-b}

4. bignum mult (bignum a, bignum b);

bignum a, b, totmult;

{P} ≡ {a.val != NULL AND b.val != NULL AND b.tam>0 AND a.tam>0 AND 0<=a.sign<=1 AND 0<=b.sign<=1 AND (∀i: 0<=i<a.tam: 0<=a.val[i]<=9) AND (∀i: 0<=i<b.tam: 0<=b.val[i]<=9)}

S ≡ mult

{Q} ≡ {totmult.tam>0 AND 0<=totmult.sign<=1 AND totmult.val!=NULL AND (∀i: 0<=i<totmult.tam: 0<=totmult.val[i]=9) AND totmult==a*b}

5. bignum remainder (bignum a, bignum b);

bignum a, b, remainder;

{P} ≡ {a.val != NULL AND n.val != NULL AND n.tam>0 AND a.tam>0 AND 0<=a.sign<=1 AND 0<=n.sign<=1 AND (∀i: 0<=i<a.tam: 0<=a.val[i]<=9) AND (∀i: 0<=i<n.tam: 0<=b.val[i]<=9) AND (n.tam != 1 OR n.val[0] != 0)}

S ≡ remainder1

{Q} ≡ {remainder.tam>0 AND 0<=remainder.sign<=1 AND remainder.val!=NULL AND (∀i: 0<=i<remainder.tam: 0<=remainder.val[i]=9) AND remainder==a%n}

6. bignum fact (bignum n);

bignum n, factorial;

{P} ≡ {n.val != NULL AND n.tam>0 AND n.sign==0 AND
(∀i: 0<=i<n.tam: 0<=n.val[i]<=9)}

S ≡ fact

{Q} ≡ {factorial.tam>0 AND factorial.sign==0 AND factorial.val!=NULL AND
(∀i: 0<=i<factorial.tam: 0<=factorial.val[i]=9) AND factorial==n!}

7. bignum multmod (bignum a, bignum b, bignum n);

bignum a, b, n, result;

{P} ≡ {a.val != NULL AND b.val != NULL AND n.val!=NULL AND
a.tam>0 AND b.tam>0 AND n.tam>0 AND 0<=a.sign<=1 AND 0<=b.sign<=1 AND
n.sign==0 AND (∀i: 0<=i<a.tam: 0<=a.val[i]<=9) AND
(∀i: 0<=i<b.tam: 0<=b.val[i]<=9) AND (∀i: 0<=i<n.tam: 0<=n.val[i]<=9) AND
(n.tam != 1 OR n.val[0] != 0)}

S ≡ multmod

{Q} ≡ {result.tam>0 AND 0<=result.sign<=1 AND result.val!=NULL AND
(∀i: 0<=i<result.tam: 0<=result.val[i]=9) AND result==(a*b)%n}

8. int comparar (bignum a, bignum b);

bignum a, b;

int mayor;

{P} ≡ {a.val != NULL AND b.val != NULL AND b.tam>0 AND a.tam>0 AND
0<=a.sign<=1 AND 0<=b.sign<=1 AND (∀i: 0<=i<a.tam: 0<=a.val[i]<=9) AND
(∀i: 0<=i<b.tam: 0<=b.val[i]<=9)}

S ≡ comparer

{Q} ≡ {mayor==-1 OR mayor==0 OR mayor==1 AND (a<b => mayor==-1) AND
(a==b => mayor==0) AND (a>b => mayor==1)}

9. void imprimir (bignum resultado);

bignum resultado;

{P} ≡ {resultado.val != NULL AND resultado.tam>0 AND 0<=resultado.sign<=1 AND (∀i: 0<=i<resultado.tam: 0<=resultado.val[i]<=9)}

S ≡ imprimir

{Q} ≡ {TRUE}