Bai toan tim uoc chung lon nhat cua 2 so:

Cho a, b la 2 so nguyen duong, a > b, tim uoc chung lon nhat cua a va b?

gcd(a, b) = gcd(a – b, b)

gcd(2021, 15) = gcd(2006, 15) = gcd(1991, 15) =

gcd(a, b) = gcd (b, a mod b)

gcd(2021, 15) = gcd(15, 11) = gcd(11, 4) = gcd (4, 3) = gcd(3, 1) = gcd(1, 0) = 1 Euclide

fun gcd (m:int, 0):int = m

| gcd (0, n:int):int = n

| gcd (m:int, n:int):int =

if m>n then gcd (m-n, n) else gcd (m,n-m);

fun gcd (m:int, 0):int = m

| gcd (0, n:int):int = n

| gcd (m:int, n:int):int =

if m>n then gcd (m mod n, n) else gcd (m, n mod m)

%%%%%%%%%%%%%%%%%%%%%%%%%

Bai toan tinh luy thua:

Cho b là 1 so va so nguyen duong e, n, tinh

e = 1000

b = 3, 2 = 200

p = 1;

for i = 1 to 1000

p = p\*b mod 200

31000 mod 200 =

3500

3250

3125

3.(360)

1000 = 2k, k = log2 1000 < log2 210 = 10

Log2 106  = 6.log2 10 = 6\*3.3 = 21

be mod n =

fun square (x) = x \* x;

fun mult (a, x) = a \* x;

fun exponent(a,0) = 1 | exponent(a,n) = mult(a,exponent(a,n-1));

fun fast\_exp (a, 0) = 1

| fast\_exp (a, n) =

if n mod 2 = 0 then

square (fast\_exp (a, n div 2))

else

mult (a, fast\_exp (a, n-1))

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

Tim giao dich khong hop le:

Cho

- Danh sach cac tai khoan hop le - goi la White list

- Cho danh sach cac giao dich trong 1 ngay nao do Transactions gom ma tai khoan giao dich

(so tien giao dich, ....)

Tim:

- Danh sach cac giao dich hop le, tuc la cac giao dich co ma tai khoan nam trong danh sach cac tai khoan hop le White list.

Log2 106 = 6. Log2 10 = 6.3.2 = 21

%%%%%%%%%%%%%%%%%%%%%%%%

fun insert(less,x,nil) = [x]

| insert(less,x,y::ys) = if less(x,y) then x::y::ys

else y::insert(less,x,ys);

fun sort(less,nil) = nil

| sort(less,x::xs) = insert(less,x,sort(less,xs));

fun less(x,y) = x < y;

sort(less,[9,5,8,2,7,1]);

fun bigger(x, y) = x > y;

sort(bigger, [5,9,2,6,1,8,3]);

fun less(x:string,y) = x < y;

sort(less,["quan","yen","binh"]);

-------------------

\*\*\*qsort

;

fun qsort nil = nil

| qsort [singleton] = [singleton]

| qsort (first::rest) =

let

val (smalls, bigs) =

partition(first,rest)

in

qsort(smalls) @ [first] @ qsort(bigs)

end

qsort [4,2,5,1,8,6,9,7];

%%%%%%%%%%%%%%%%%%%%%%%%%%%

fun equal(x:int,y) = x=y

fun search(x, nil) = false | search(x, fst::rest) = if equal(x,fst) then true else search(x,rest)

datatype ’a tree = Niltree |

Maketree of ’a \* (’a tree) \* (’a tree)

fun insert (new:int) Niltree =

Maketree(new,Niltree,Niltree)

| insert new (Maketree (root,l,r)) =

if new < root

then Maketree (root,(insert new l),r)

else Maketree (root,l,(insert new r))

fun buildtree [] = Niltree

| buildtree (fst :: rest) =

insert fst (buildtree rest)

fun find (elt:int) Niltree = false

| find elt (Maketree (root,left,right)) =

if elt = root

then true

else if elt < root

then find elt left

else find elt right (\* elt > root \*)

val cay = buildtree [9,1,10,3,7,8,4]

find 3 cay

fun bsearch elt list =

find elt (buildtree list);

val x = bsearch 7 [9,1,10,3,7,5,4,8];