COMP108
Data Structures and Algorithms

Pseudo code (Part II)

Professor Prudence Wong

pwong@liverpool.ac.uk

2020-21

Developing pseudo code

- Suppose x & y are both integers.
- Write a while loop to output all factors of x which are not factors of y.
- Examples of expected results:

x	y	factors of x	output
6	3	~~1~~, 2, ~~3~~, 6	2, 6
30	9	~~1~~, 2, ~~3~~, 5, 6, 10, 15, 30	2, 5, 6, 10, 15, 30
3	6	~~1~~, ~~3~~	-

Developing pseudo code

- Suppose x & y are both integers.
- Write a while loop to output all factors of x which are not factors of y.
- Examples of expected results:

x	y	factors of x	output
6	3	1, 2, 3, 6	2, 6
30	9	1, 2, 3, 5, 6, 10, 15, 30	2, 5, 6, 10, 15, 30
3	6	1, 3	-

- Skeleton:

Developing pseudo code

- Suppose x & y are both integers.
- Write a while loop to output all factors of x which are not factors of y.
- Examples of expected results:

x	y	factors of x	output
6	3	1, 2, 3, 6	2, 6
30	9	1, 2, 3, 5, 6, 10, 15, 30	2, 5, 6, 10, 15, 30
3	6	1, 3	-

- Skeleton:

```
i ← ?
```

Developing pseudo code

- Suppose x & y are both integers.
- Write a while loop to output all factors of x which are not factors of y.
- Examples of expected results:

x	y	factors of x	output
6	3	1, 2, 3, 6	2, 6
30	9	1, 2, 3, 5, 6, 10, 15, 30	2, 5, 6, 10, 15, 30
3	6	1, 3	-

- Skeleton:

```
i ← ?
while ???? do
begin


  i ← ??
end
```

Developing pseudo code

- Suppose x & y are both integers.
- Write a while loop to output all factors of x which are not factors of y.
- Examples of expected results:

x	y	factors of x	output
6	3	1, 2, 3, 6	2, 6
30	9	1, 2, 3, 5, 6, 10, 15, 30	2, 5, 6, 10, 15, 30
3	6	1, 3	-

- Skeleton:

```
i ← ?
while ???? do
begin
    if ??? then
        output ???
    i ← ??
end
```

Factors of x but not factors of y

Steps:

- Factor of x must be between 1 and x

```
i = 1
while ( $i \le x$ )
begin
    $i \leftarrow i+1$
end
```

- If x is divisible by i, then i is a factor of x
- If y is not divisible by i, then i is not a factor of y
- We need both conditions to hold and then we output i

Factors of x but not factors of y

Steps:

- Factor of x must be between 1 and x

```
i ← 1


begin
   i ← i+1
end
```

- If x is divisible by i, then i is a factor of x

- If y is not divisible by i, then i is not a factor of y

- We need both conditions to hold and then we output i

Factors of x but not factors of y

Steps:

- Factor of x must be between 1 and x

```
i ← 1
while i ≤ x do
begin
    i ← i+1
end
```

- If x is divisible by i, then i is a factor of x
- If y is not divisible by i, then i is not a factor of y
- We need both conditions to hold and then we output i

Factors of x but not factors of y

Steps:

- Factor of x must be between 1 and x

```
i $\leftarrow$ 1
while i $\leq$ x do
begin
    i $\leftarrow$ i+1
end
```

- If x is divisible by i, then i is a factor of x

```
if x%i == 0 then
```

- If y is not divisible by i, then i is not a factor of y

- We need both conditions to hold and then we output i

Factors of x but not factors of y

Steps:

- Factor of x must be between 1 and x

```
i $\leftarrow$ 1
while i $\leq$ x do
begin
    i $\leftarrow$ i+1
end
```

- If x is divisible by i, then i is a factor of x

```
if x%i == 0 then
```

- If y is not divisible by i, then i is not a factor of y

```
if y%i $\neq$ 0 then
```

- We need both conditions to hold and then we output i

Factors of x but not factors of y

Steps:

- Factor of x must be between 1 and x

```
i $\leftarrow$ 1
while i $\leq$ x do
begin
    i $\leftarrow$ i+1
end
```

- If x is divisible by i, then i is a factor of x

```
if x%i == 0 then
```

- If y is not divisible by i, then i is not a factor of y

```
if y%i $\neq$ 0 then
```

- We need both conditions to hold and then we output i

```
if x%i == 0 AND y%i $\neq$ 0 then
    output i
```

Factors of x but not factors of y

Summarizing:

Factors of x but not factors of y

Summarizing:

```
i $\leftarrow$ 1
while i $\leq$ x do
begin


  i $\leftarrow$ i+1
end
```

Factors of x but not factors of y

Summarizing:

```
i $\leftarrow$ 1
while i $\leq$ x do
begin
    if x%i == 0 AND y%i $\neq$ 0 then
        output i
    i $\leftarrow$ i+1
end
```

Finding Lowest Common Multiple (LCM)

Suppose x < y & both are integers.
Write a while loop to output the lowest common multiple lcm of x and y, i.e., the smallest number that is divisible by x and divisible by y.

Finding Lowest Common Multiple (LCM)

Suppose x < y & both are integers.
Write a while loop to output the lowest common multiple lcm of x and y, i.e., the smallest number that is divisible by x and divisible by y.

- So we want
  - lcm%x to be 0
  - lcm%y to be 0
  - lcm to be as small as possible

Finding Lowest Common Multiple (LCM)

Suppose x < y & both are integers.

Write a while loop to output the lowest common multiple lcm of x and y, i.e., the smallest number that is divisible by x and divisible by y.

- So we want
  - lcm%x to be 0
  - lcm%y to be 0
  - lcm to be as small as possible
- lcm cannot be smaller than y and would not be larger than x*y.

lcm → start from y

+1, +1, +1

→ until x*y

Finding Lowest Common Multiple (LCM)

Suppose x < y & both are integers.

Write a while loop to output the lowest common multiple lcm of x and y, i.e., the smallest number that is divisible by x and divisible by y.

- So we want
  - lcm%x to be 0
  - lcm%y to be 0
  - lcm to be as small as possible
- lcm cannot be smaller than y and would not be larger than x*y.
  - we can start lcm from y, increase by 1 every time, and then check if lcm is multiple of both x and y

Finding Lowest Common Multiple (LCM)

Suppose x < y & both are integers.

Write a while loop to output the lowest common multiple lcm of x and y, i.e., the smallest number that is divisible by x and divisible by y.

- So we want
  - lcm%x to be 0
  - lcm%y to be 0
  - lcm to be as small as possible
- lcm cannot be smaller than y and would not be larger than x*y.
  - we can start lcm from y, increase by 1 every time, and then check if lcm is multiple of both x and y
  - we can start lcm from y, increase by y every time, and then check if lcm is multiple of x (we don't need to check if lcm is multiple of y because it is already)

lcm → start from y

+y , +y , +y

Finding Lowest Common Multiple (LCM)

Suppose x < y & both are integers.
Write a while loop to output the lowest common multiple lcm of x and y, i.e., the smallest number that is divisible by x and divisible by y.

- So we want
    - lcm%x to be 0
    - lcm%y to be 0
    - lcm to be as small as possible
- lcm cannot be smaller than y and would not be larger than x*y.
    - we can start lcm from y, increase by 1 every time, and then check if lcm is multiple of both x and y
    - we can start lcm from y, increase by y every time, and then check if lcm is multiple of x (we don't need to check if lcm is multiple of y because it is already)
    - once we find the smallest one, we should stop

LCM

```
lcm ← _____, found ← false
while lcm ≤ _______ AND found ≠ true do
begin
    if ____________________________ then
        found ← true
    else lcm ← lcm + ____
end
output lcm
```

found ← ~~false~~ true

continue   stop

found ≠ true

```
lcm ← _____, found ← false
while lcm ≤ _______ AND found ≠ true do
begin
    if ____________ then
        found ← true
    else lcm ← lcm + ____
end
output lcm
```

LCM

```
lcm ← __y__ , found ← false
while lcm ≤ __x*y__ AND found ≠ true do
begin
    if ______________________ then
        found ← true
    else lcm ← lcm + __1__
end
output lcm
```

```
lcm ← _____ , found ← false
while lcm ≤ _______ AND found ≠ true do
begin
    if ___________ then
        found ← true
    else lcm ← lcm + ____
end
output lcm
```

lcm % x == 0

lcm is multiple of x

AND

lcm is multiple of y

lcm % y == 0

LCM

```
lcm ← __y__, found ← false
while lcm ≤ __x*y__ AND found ≠ true do
begin
    if __lcm%x==0 AND lcm%y==0__ then
        found ← true
    else lcm ← lcm + __1__
end
output lcm
```

```
lcm ← _____, found ← false
while lcm ≤ _______ AND found ≠ true do
begin
    if ____________ then
        found ← true
    else lcm ← lcm + ____
end
output lcm
```

LCM

```
lcm ← __y__, found ← false
while lcm ≤ __x*y__ AND found ≠ true do
begin
    if __lcm%x==0 AND lcm%y==0__ then
        found ← true
    else lcm ← lcm + __1__
end
output lcm
```

```
lcm ← __y__, found ← false
while lcm ≤ __x*y__ AND found ≠ true do          lcm % x == 0
begin
    if ______ then
        found ← true
    else lcm ← lcm + ______     y
end
output lcm
```

LCM

```
lcm ← __y__, found ← false
while lcm ≤ __x*y__ AND found ≠ true do
begin
    if __lcm%x==0 AND lcm%y==0__ then
        found ← true
    else lcm ← lcm + __1__
end
output lcm
```

```
lcm ← __y__, found ← false
while lcm ≤ __x*y__ AND found ≠ true do
begin
    if __lcm%x==0__ then
        found ← true
    else lcm ← lcm + __y__
end
output lcm
```

LCM

```
lcm ← y , found ← false
while lcm ≤ x*y AND found ≠ true do
begin
    if lcm%x==0 AND lcm%y==0 then
        found ← true
    else lcm ← lcm + 1
end
output lcm
```

```
lcm ← y , found ← false
while lcm ≤ x*y AND found ≠ true do
begin
    if lcm%x==0 then
        found ← true
    else lcm ← lcm + y
end
output lcm
```

break? not a good practice

Questions

- Is the condition ``lcm ≤ x*y'' necessary? Not necessary
- Why do we need to use the flag variable `found`?
- What happens if we remove the keyword ``else''?

Approach 1: incorrect answer lcm + 1

Approach 2: incorrect answer lcm + y

Summary: Developing pseudo code

Next: Exercises on pseudo code

For note taking