CMPS 10 10-8-09

EXERCISE:

Assume mult. is primitive, but exponentiation is NOT.

Input: Two non-negative ints. a, b

Output: a^b

This will not work!

1.) get a, b
2.) print a^b
3.) stop

<u>EXERCISE</u>:

Assume mult. is Primitive, but factorial is not.

$$\begin{cases} n! = 1 \cdot 2 \cdot 3 \cdots n & \text{if } n \geq 1 \\ 0! = 1 \end{cases}$$

write an algorithm which takes any $n \geq 0$, then prints $n!$

PROBLEM: PATTERN MATCHING.

INPUT: $1 \le m \le n$

text: $T_1 T_2 T_3 \ldots\ldots T_n$

Pattern: $P_1 P_2 \ldots P_m$

OUTPUT: every occurance of Pattern in text.

EX. $n = 18$, $m = 2$

text: 'to be or not to be'

Pattern: 'be'

Answer: 4, 17

Ex. $n=10$, $m=3$

text: 'xxxaaaaaxx'

Pattern: 'aaa'

answer: 4, 5, 6

Ex. $n=7$, $m=3$

check match at position

$T_1\ T_2\ T_3\ T_4\ T_5\ T_6\ T_7$
$P_1\ P_2\ P_3$
$i=1$

$T_1\ T_2\ T_3\ T_4\ T_5\ T_6\ T_7$
$\longrightarrow P_1\ P_2\ P_3$
$i=2$

$T_1\ T_2\ T_3\ T_4\ T_5\ T_6\ T_7$
$\longrightarrow P_1\ P_2\ P_3$
$i=3$

$T_1\ T_2\ T_3\ T_4\ T_5\ T_6\ T_7$
$\longrightarrow P_1\ P_2\ P_3$
$i=4$

$T_1\ T_2\ T_3\ T_4\ T_5\ T_6\ T_7$
$\longrightarrow P_1\ P_2\ P_3$
$i=5$

POSSIBLE ANSWERS: $1 \le i \le n-m+1$

INPUT: n, m s.t. $1 \le m \le n$

$T_1 \ldots T_n$, $P_1 \ldots P_m$

OUTPUT: All indicies i such that $P_1 \ldots P_m$ matches $T_i \ldots T_{i+m-1}$ where $1 \le i \le n-m+1$.

Pattern Match

```
i ← 1
while i ≤ n-m+1
    j ← 1
    match ← true
    while j ≤ m and match
        if P_j ≠ T_{i+j-1}
            match ← false
        else
            j ← j+1
    if match
        Print 'match found at position' i
    i ← i+1
stop
```

EXERCISE:

text: 'hand the band to Randy and I'

Pattern: 'and'

Pattern: '_and_'

Chapter 3 Attributes of Algorithms

- Correctness
- Clarity
- Elegance
- Efficiency

Clarity: Recall Seq. Search

10.) if not found

or 10.) if found = false

or 10.) if $i > n$

Elegance:

Ex.

1.) get n

2.) sum $\leftarrow$ 0

3.) i $\leftarrow$ 1

4.) while i $\leq$ n

5.) sum $\leftarrow$ sum + i

6.) i $\leftarrow$ i+1

7.) print sum

8.) stop

Prints out:

$1+2+3+\ldots\ldots+(n-1)+n$

Equivalent Algorithm:

1.) get n

2.) Print $\frac{n(n+1)}{2}$

3.) stop

Formula:

$$1+2+3+\cdots\cdots+n = \frac{n(n+1)}{2}$$

Proof:

let

$$S = 1+2+3+\cdots\cdots+(n-2)+(n-1)+n$$

then

$$S = n+(n-1)+(n-2)+\cdots+3+2+1$$

so

$$S+S = (n+1)+(n+1)+(n+1)+\cdots+(n+1)+(n+1)+(n+1)$$

So

$$2S = n(n+1)$$

$$\therefore \quad S = \frac{n(n+1)}{2} \quad .$$

///

EXERCISE:

Write an algorithm which takes

INPUT: integer n

OUTPUT: sum of first n odd positive integers:

$$1 + 3 + 5 + \cdots + (2n-1)$$

Do this in two ways

a.) with a loop.

b.) Derive a formula similar to Gauss'

Efficiency: (time)

- we count # of operations performed, not # of seconds
- measures of runtime

  Best case

  Worst case

  Average case

- Dont count all operations. Pick a Basic (or Barometer) operation, and count it.

Recall Sequential Search:

```
1.) get n, a1, ..., an, target
2.) i ← 1
3.) found ← false
4.) while i ≤ n and not found
5.)       if ai = target      ⇐ BASIC OP.
6.)             found ← true
7.)       else
8.)             i ← i+1
9.) if not found
10.)        i ← 0
11.) Print i
12.) stop.
```