Digital World (2018)

Week 5, S3: Calendar Program; Recursion

Chris Poskitt

SINGAPORE UNIVERSITY OF
TECHNOLOGY AND DESIGN

construct_cal_month: the rough idea

initialise

---

result

["January"]

construct_cal_month: the rough idea

initialise

day_of_the_month ——> 1
day_of_the_week ——> 0

---

result

["January"]

construct_cal_month: the rough idea

initialise

day_of_the_month ——> 1
day_of_the_week ——> 0

---

while *day_of_the_month* <= *num_days_in_month*

---

result

["January"]

construct_cal_month: the rough idea

initialise

---

while *day_of_the_month* <= *num_days_in_month*

day_of_the_month ——> 1

day_of_the_week ——> 0

---

result

["January"]

construct_cal_month: the rough idea

initialise

---

while *day_of_the_month* <= *num_days_in_month*

day_of_the_month ——> 1

day_of_the_week ——> 0

“”

---

result

[“January”]

construct_cal_month: the rough idea

initialise

---

while *day_of_the_month* <= *num_days_in_month*

day_of_the_month ——> 2

day_of_the_week ——> 1

“ 1”

---

result

[“January”]

construct_cal_month: the rough idea

initialise

---

while *day_of_the_month* <= *num_days_in_month*

day_of_the_month ——> 3

day_of_the_week ——> 2

“ 1 2”

---

result

[“January”]

construct_cal_month: the rough idea

initialise

---

while *day_of_the_month* <= *num_days_in_month*

day_of_the_month ——> 4

day_of_the_week ——> 3

“ 1 2 3”

---

result

[“January”]

construct_cal_month: the rough idea

initialise

---

while *day_of_the_month* <= *num_days_in_month*

day_of_the_month ——> 5

day_of_the_week ——> 4

“ 1 2 3 4”

---

result

[“January”]

construct_cal_month: the rough idea

initialise

---

while *day_of_the_month* <= *num_days_in_month*

day_of_the_month ——> 6

day_of_the_week ——> 5

“ 1 2 3 4 5”

---

result

[“January”]

construct_cal_month: the rough idea

initialise

---

while *day_of_the_month* <= *num_days_in_month*

day_of_the_month ——> 7

day_of_the_week ——> 6

“ 1 2 3 4 5 6”

---

result

[“January”]

construct_cal_month: the rough idea

initialise

---

while *day_of_the_month* <= *num_days_in_month*

day_of_the_month ——> 8

day_of_the_week ——> 7

“ 1 2 3 4 5 6 7”

---

result

[“January”]

construct_cal_month: the rough idea

initialise

---

while *day_of_the_month* <= *num_days_in_month*

day_of_the_month ——> 8
day_of_the_week ——> 7

end of the week!

“ 1 2 3 4 5 6 7”

---

result

[“January”]

construct_cal_month: the rough idea

initialise

---

while *day_of_the_month* <= *num_days_in_month*

day_of_the_month ——> 8
day_of_the_week ——> 7

end of the week!

---

result

["January"] " 1 2 3 4 5 6 7"

construct_cal_month: the rough idea

initialise

---

while *day_of_the_month* <= *num_days_in_month*

day_of_the_month ——> 8

day_of_the_week ——> 0

“”

---

result

[“January”,“ 1 2 3 4 5 6 7”]

construct_cal_month: the rough idea

initialise

---

while *day_of_the_month* <= *num_days_in_month*

day_of_the_month ——> 9

day_of_the_week ——> 1

“ 8”

---

result

[“January”,“ 1 2 3 4 5 6 7”]

construct_cal_month: the rough idea

initialise

---

while *day_of_the_month* <= *num_days_in_month*

day_of_the_month ——> 10

day_of_the_week ——> 2

“ 8 9”

---

result

[“January”,“ 1 2 3 4 5 6 7”]

construct_cal_month: the rough idea

initialise

---

while *day_of_the_month* <= *num_days_in_month*

day_of_the_month ——> 11

day_of_the_week ——> 3

“ 8 9 10”

---

result

[“January”,“ 1 2 3 4 5 6 7”]

construct_cal_month: the rough idea

initialise

---

while *day_of_the_month* <= *num_days_in_month*

day_of_the_month ——> 12

day_of_the_week ——> 4

“ 8 9 10 11”

---

result

[“January”,“ 1 2 3 4 5 6 7”]

construct_cal_month: the rough idea

initialise

---

while *day_of_the_month* <= *num_days_in_month*

day_of_the_month ——> 13

day_of_the_week ——> 5

“ 8 9 10 11 12”

---

result

[“January”,“ 1 2 3 4 5 6 7”]

construct_cal_month: the rough idea

initialise

---

while *day_of_the_month* <= *num_days_in_month*

day_of_the_month ——> 14

day_of_the_week ——> 6

" 8 9 10 11 12 13"

---

result

["January"," 1 2 3 4 5 6 7"]

construct_cal_month: the rough idea

initialise

---

while *day_of_the_month* <= *num_days_in_month*

day_of_the_month ——> 15

day_of_the_week ——> 7

“ 8 9 10 11 12 13 14”

---

result

[“January”,“ 1 2 3 4 5 6 7”]

construct_cal_month: the rough idea

initialise

---

while *day_of_the_month* <= *num_days_in_month*

day_of_the_month ——> 15

day_of_the_week ——> 0

“”

---

result

[“January”,“ 1 2 3 4 5 6 7”,“ 8 9 10 11 12 13 14”]

construct_cal_month: the rough idea

initialise

---

while *day_of_the_month* <= *num_days_in_month*

day_of_the_month ——> 16

day_of_the_week ——> 1

" 15"

---

result

["January"," 1 2 3 4 5 6 7"," 8 9 10 11 12 13 14"]

construct_cal_month: the rough idea

initialise

---

while *day_of_the_month* <= *num_days_in_month*

day_of_the_month ——> 17

day_of_the_week ——> 2

“ 15 16”

---

result

[“January”,“ 1 2 3 4 5 6 7”,“ 8 9 10 11 12 13 14”]

construct_cal_month: the rough idea

initialise

---

while *day_of_the_month* <= *num_days_in_month*

day_of_the_month ——> 17

day_of_the_week ——> 2

" 15 16" *etc.*

---

result

["January"," 1 2 3 4 5 6 7"," 8 9 10 11 12 13 14"]

construct_cal_month: the rough idea

initialise

---

result

["May"]

construct_cal_month: the rough idea

initialise

day_of_the_month ——> 1
day_of_the_week ——> 4

---

result

["May"]

construct_cal_month: the rough idea

initialise

day_of_the_month ——> 1
day_of_the_week ——> 4

---

while *day_of_the_month* <= *num_days_in_month*

---

result

["May"]

construct_cal_month: the rough idea

initialise

---

while *day_of_the_month* <= *num_days_in_month*

day_of_the_month ——> 1

day_of_the_week ——> 4

---

result

["May"]

construct_cal_month: the rough idea

initialise

---

while *day_of_the_month* <= *num_days_in_month*

day_of_the_month ——> 1

day_of_the_week ——> 4

“ ”

---

result

[“May”]

construct_cal_month: the rough idea

initialise

---

while *day_of_the_month* <= *num_days_in_month*

day_of_the_month ——> 2

day_of_the_week ——> 5

“ ”

---

result

[“May”]

construct_cal_month: the rough idea

initialise

---

while *day_of_the_month* <= *num_days_in_month*

day_of_the_month ——> 3

day_of_the_week ——> 6

“ 1 2”

---

result

[“May”]

construct_cal_month: the rough idea

initialise

---

while *day_of_the_month* <= *num_days_in_month*

day_of_the_month ——> 4

day_of_the_week ——> 7

“ 1 2 3”

---

result

[“May”]

construct_cal_month: the rough idea

initialise

---

while *day_of_the_month* <= *num_days_in_month*

day_of_the_month ——> 4
day_of_the_week ——> 7

end of the week!

“ 1 2 3”

---

result

[“May”]

construct_cal_month: the rough idea

initialise

---

while *day_of_the_month* <= *num_days_in_month*

day_of_the_month ——> 4
day_of_the_week ——> 7

end of the week!

---

result

["May"] " 1 2 3"

construct_cal_month: the rough idea

initialise

---

while *day_of_the_month* <= *num_days_in_month*

day_of_the_month ——> 4

day_of_the_week ——> 0

“”

---

result

[“May”,“ 1 2 3”]

construct_cal_month: the rough idea

initialise

---

while *day_of_the_month* <= *num_days_in_month*

day_of_the_month ——> 5

day_of_the_week ——> 1

“ 4”

---

result

[“May”,“ 1 2 3”]

construct_cal_month: the rough idea

initialise

---

while *day_of_the_month* <= *num_days_in_month*

day_of_the_month ——> 6

day_of_the_week ——> 2

“ 4 5”

---

result

[“May”,“ 1 2 3”]

construct_cal_month: the rough idea

initialise

---

while *day_of_the_month* <= *num_days_in_month*

day_of_the_month ——> 7

day_of_the_week ——> 3

“ 4 5 6”

---

result

[“May”,“ 1 2 3”]

construct_cal_month: the rough idea

initialise

---

while *day_of_the_month* <= *num_days_in_month*

day_of_the_month ——> 8

day_of_the_week ——> 4

“ 4 5 6 7”

---

result

[“May”,“ 1 2 3”]

construct_cal_month: the rough idea

initialise

---

while *day_of_the_month* <= *num_days_in_month*

day_of_the_month ——> 9

day_of_the_week ——> 5

“ 4 5 6 7 8”

---

result

[“May”,“ 1 2 3”]

construct_cal_month: the rough idea

initialise

---

while *day_of_the_month* <= *num_days_in_month*

day_of_the_month ——> 10

day_of_the_week ——> 6

“ 4 5 6 7 8 9”

---

result

[“May”,“ 1 2 3”]

construct_cal_month: the rough idea

initialise

---

while *day_of_the_month* <= *num_days_in_month*

day_of_the_month ——> 11

day_of_the_week ——> 7

“ 4 5 6 7 8 9 10”

---

result

[“May”,“ 1 2 3”]

construct_cal_month: the rough idea

initialise

---

while *day_of_the_month* <= *num_days_in_month*

day_of_the_month ——> 11

day_of_the_week ——> 0

“”

---

result

[“May”,“ 1 2 3”,“ 4 5 6 7 8 9 10”]

construct_cal_month: the rough idea

initialise

---

while *day_of_the_month* <= *num_days_in_month*

day_of_the_month ——> 12

day_of_the_week ——> 1

“ 11”

---

result

[“May”,“ 1 2 3”,“ 4 5 6 7 8 9 10”]

construct_cal_month: the rough idea

initialise

---

while *day_of_the_month* <= *num_days_in_month*

day_of_the_month ——> 12

day_of_the_week ——> 1

" 11" *etc.*

---

result

["May"," 1 2 3"," 4 5 6 7 8 9 10"]

Aside: what does **PHP** stand for?

Aside: what does **PHP** stand for?

PHP: Hypertext Preprocessor

Aside: what does **PHP** stand for?

PHP: Hypertext Preprocessor

PHP: Hypertext Preprocessor: Hypertext Preprocessor

Aside: what does **PHP** stand for?

PHP: Hypertext Preprocessor

PHP: Hypertext Preprocessor: Hypertext Preprocessor

PHP: Hypertext Preprocessor: Hypertext Preprocessor: Hypertext Preprocessor

Recursive functions

- functions that call themselves are called recursive

- idea: solution to a problem depends on solutions to smaller instances of the same problem

- to solve a problem recursively, define:

  => *the base case(s), covering the "simplest" case(s), for which no recursion is needed (e.g. return a constant)*

  => *the general case, in which recursion is used to bring the computation "closer" to the base case(s)*

- (best understood from examples!)

Recursion example: factorial

- write a function that computers the factorial n!
- what is the base case? and the recursive case?

Recursion example: factorial

- write a function that computers the factorial n!
- what is the base case? and the recursive case?

0! = 1

Recursion example: factorial

- write a function that computers the factorial n!
- what is the base case? and the recursive case?

0! = 1

$$n! = n \times (n-1)!$$

(for n > 0)

Recursion example: factorial

- write a function that computers the factorial n!
- what is the base case? and the recursive case?

$0! = 1$

$n! = n \times (n-1)!$

(for n > 0)

4! = 4 x 3!

Recursion example: factorial

- write a function that computers the factorial n!

- what is the base case? and the recursive case?

0! = 1

$$n! = n \times (n-1)!$$

(for n > 0)

4! = 4 x (3 x 2!)

Recursion example: factorial

- write a function that computers the factorial n!
- what is the base case? and the recursive case?

0! = 1

$$n! = n \times (n-1)!$$

(for n > 0)

4! = 4 x (3 x (2 x 1!))

Recursion example: factorial

- write a function that computers the factorial n!

- what is the base case? and the recursive case?

$0! = 1$

$n! = n \times (n-1)!$

(for n > 0)

4! = 4 x (3 x (2 x (1 x 0!)))

Recursion example: factorial

- write a function that computers the factorial n!
- what is the base case? and the recursive case?

$0! = 1$

$n! = n \times (n-1)!$

(for n > 0)

4! = 4 x (3 x (2 x (1 x 1)))

Recursion example: factorial

- write a function that computers the factorial n!

- what is the base case? and the recursive case?

0! = 1

$$n! = n \times (n-1)!$$

(for n > 0)

4! = 4 x (3 x (2 x 1))

Recursion example: factorial

- write a function that computers the factorial n!
- what is the base case? and the recursive case?

0! = 1

$$n! = n \times (n-1)!$$

(for n > 0)

4! = 4 x (3 x 2)

Recursion example: factorial

- write a function that computers the factorial n!

- what is the base case? and the recursive case?

0! = 1

$$n! = n \times (n-1)!$$

(for n > 0)

4! = 4 x 6

Recursion example: factorial

- write a function that computers the factorial n!
- what is the base case? and the recursive case?

0! = 1

$$n! = n \times (n-1)!$$

(for n > 0)

4! = 24

Reminder: TA Consultation Sessions

[DW] A reminder of the general consultation sessions!

A reminder of the consultation sessions open to all of you on the following dates.

Week	Date	Day	Time	Venue	Address
5	23 Feb 2018	Friday	1400 - 1700	ThinkTank 11	1.503
6	28 Feb 2018	Wednesday	1400 - 1700	ThinkTank 13	1.508

The TAs are ready to help you!

Reminder: Prepare for 1D Final Project

- please read: http://tiny.cc/1d_final_project

- open project within scope of "*smart green housing and neighbourhood*"

- solution must have a GUI, and must use either the Thymio or a Raspberry Pi; each group has a budget of $50

- **week 6 deadline:** presentation slides (max. 5 slides / 5 mins); *must* submit to eDimension before Session 3