# COMP20005 Workshop Week 10

| | |
|---|---|
| 1 | Structures Revisited & Ex 8.1 |
| 2 | Case Study: Polygons (Ex. 8.2 & 8.3) |
| 3 | Time for fun: quiz |
| 4 | Assignment2 Q&A |
| 5 | Lab: Work on ass2, and/or:<br>• explore exercise 8.9 in github<br>• fully implement ex 8.2-8.4 using use data file polys.txt in github for testing |

# Structures Revisited

**Ex 8.1 (simplified)**

Supposing:

`NAMELEN is 40`

2) Draw a diagram for `name`.

2) Write input statement for `name`

3) How many bytes used:

- `name`
- `date`

```
typedef char namestr[NAMELEN+1];
typedef struct {
    namestr given, others, family;
} name_t;
typedef struct{
    int dd, mm, yyyy;
} date_t;

name_t name;
date_t date;
```

# Structures Revisited

**Ex 8.1 (simplified)**

Supposing:

`NAMELEN` and `STUDSMAX` are 40 and 50000

1) Draw a diagram for `stud`.

2) How many bytes used:

- `stud`
- `unimelb`

```
typedef struct {
    name_t name;
    int id;
    date_t dob;
    // int ns;
    // subject_t subs[SUBSMAX];
} student_t;
student_t stud;
student_t unimelb[STUDSMAX];
```

# pointers to struct

```
typedef struct {
   name_t name;
   int id;
   date_t dob;
   // int ns;
   // subject_t subs[SUBSMAX];
} student_t;

student_t stud;
student_t *ps;
ps = &stud;
```

member `id` of `stud` can be written as `stud.id.` It can also be written as `(*ps).id` and `ps->id`

`ps->` is a shorthand for `(*ps).`

# pointers to struct

```
typedef struct {
   name_t name;
   int id;
   date_t dob;
   // int ns;
   // subject_t subs[SUBSMAX];
} student_t;

student_t stud;
student_t *ps;
ps = &stud;

scanf("%s%s%s%d%d%d%d", stud.name.given,...,
       &stud.id, &stud.dob.dd, &stud.dob.mm,...);

scanf("%s%s%s%d%d%d%d", ps->name.given,...,
       &ps->id, &ps->dob.dd, &ps->dob.mm,...);
```

## Structures revisited

Write a function header and function call for inputting one student record into variable stud.

How many bytes are passed:

- (at the beginning) from the caller to the function, and
- (at the end) from the function to the caller.

# write function to read a student

```
typedef struct {
   name_t name;
   int id;
   date_t dob;
} student_t;


?? read_stud( ??? ) {

   scanf("%s%s%s %d %d/%d/%d",




}
```

## Structures: function for input, version 1

```
student_t read_stud() {
  student_t s;
  scanf("%s%s%s %d %d/%d/%d", s.name.given,
          s.name.others, s.name.family,
          &s.id,
          &s.dob.dd, &s.dob.mm, &s.dob.yyyy);
  return s;
}
```

*How many bytes copied with the function call?*

```
student_t stud;
stud= read_stud();
```

## Structures: function for input, version 2

```
void read_stud(student_t &ps) {
   name_t *pn= &ps->name; // for convenience
   scanf(“%s%s%s %d %d/%d/%d”, ps->name.given,
      pn->others, pn->family,
      &ps->id,
      &ps->dob.dd, &ps->dob.mm, &ps->dob.yyyy);
}
```

```
How many bytes copied with the function call?
student_t stud;
readtime(&stud);
```

## Structures revisited: a rule

Don't use a `struct` for a function argument, use a `pointer to struct` instead.

## Quiz 1

*Which of the following accesses a variable in structure* `b`*?*

A. `b->var`

B. `b.var`

C. `*b.var`

D. `b[var]`

# Quiz 2

*Which of the following accesses a variable in a pointer to a structure* `b`*?*

A. `b->var`

B. `b.var`

C. `*b.var`

D. `(*b).var`

## Special Quiz

Your tutor’s name is:

A. `You-Know-Who`
B. `Anh Vo`
C. `Alistair Moffat`
D. `Michael Turnbull`

# Case Study: Polygons (Ex 8.2-8.4)

*Suppose that a closed polygon is represented as a sequence of points in two dimensions. Give suitable declarations for a type* `poly_t`*, assuming that a polygon has no more than 100 points.*

*a) Build a data file* `polys.txt` *with content:*

```
3  0 0  3 0  0 4
4  5 0  6 0  6 1  5 1
```

*which represent a triangle and a square.*

*b) Write a program that includes the following functions that*

- *(i) reads a poly from* `stdin`
- *(ii) returns the length of the perimeter of a polygon (ex 8.3).*
- *(iii) returns the area of a polygon (ex 8.4) .*
- *(iv) return distance between the centroids of two polygon.*

*Test these functions using data from* `polys.txt`*.*

## ass1 review: a sample solution

## assignment 2: new in rubric

- avoidance of structs (eg, using multiple 2d arrays), -1.0;
- avoidance of struct pointers (eg, using whole-struct arguments), -0.5;
- inappropriate or over-complex structs, -0.5;
- other abuses of structs, -0.5;

## assignment 2: data

```
label    xloc    yloc    liters   rootrad
A        14.8    23.8    185000   8.0
G        18.8    28.1    208000   7.0
F        24.1    22.2    310000   6.2
C        35.3    19.9    280000   7.3
E        16.5    10.5    150000   4.2
```

So:

- exactly one header line
- one data line per tree, data are well-formatted
- the tree label is just a **`char`**

## assignment 2: stage 1

```
label    xloc    yloc    liters  rootrad
A        14.8    23.8    185000  8.0
G        18.8    28.1    208000  7.0
F        24.1    22.2    310000  6.2
C        35.3    19.9    280000  7.3
E        16.5    10.5    150000  4.2
```

```
S1: total data lines   =     5 trees
S1: total water needed = 1.133 megaliters per year
```

Notes:

- mega = $10^6$
- try to have similar number format: right after '=' there are 6 positions in total for numbers, `xloc` needs 4 positions…

# assignment 2: stage 2

```
label    xloc    yloc
A        14.8    23.8
G        18.8    28.1
F        24.1    22.2
C        35.3    19.9
E        16.5    10.5
```

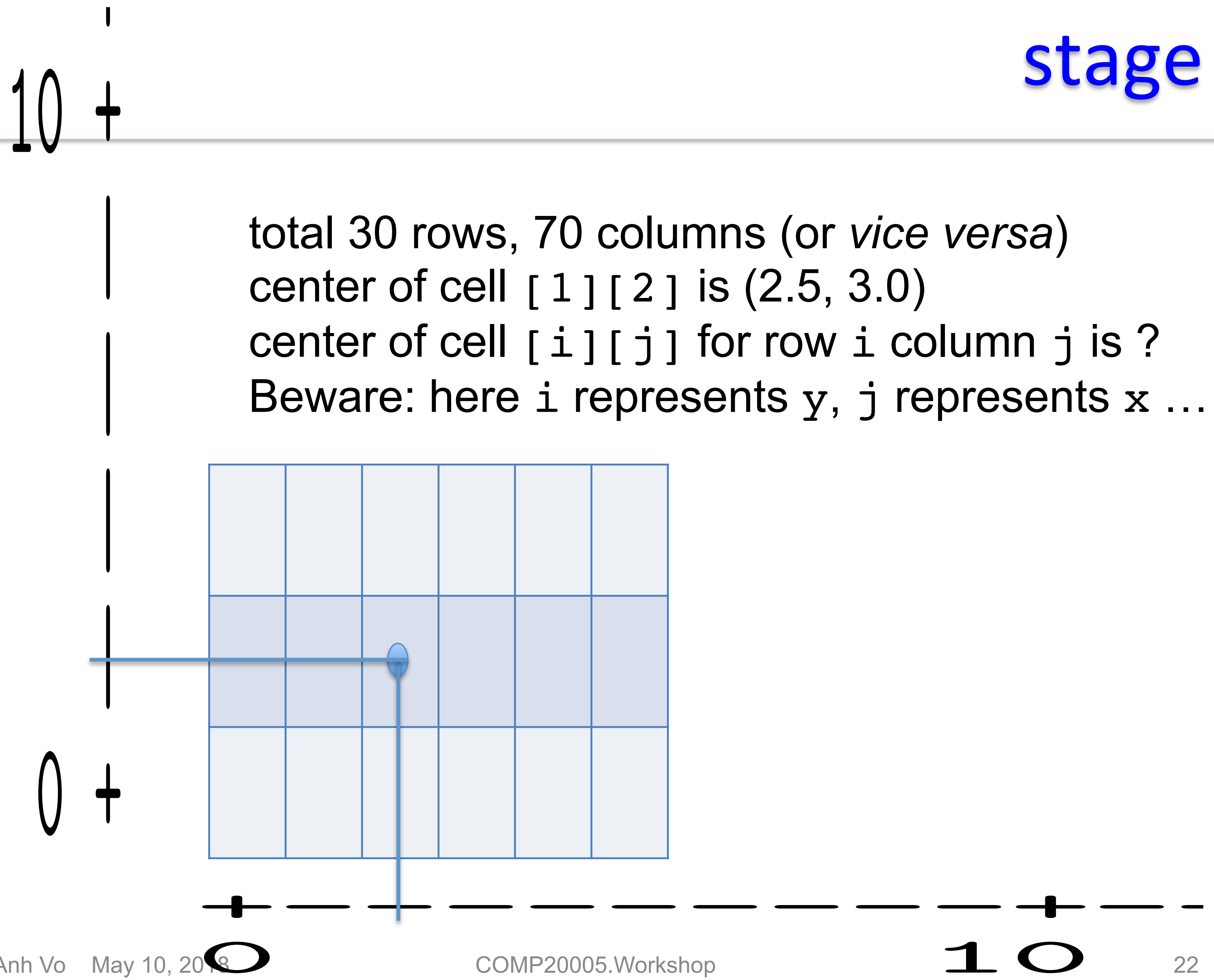
```
S2: tree A is in conflict with G F
S2: tree G is in conflict with A F
S2: tree F is in conflict with A G C
S2: tree C is in conflict with F
S2: tree E is in conflict with
```

Notes:

- `F` is in the list for `A`, and `A` is in the list for `F`
- `E` doesn't have conflicts at all, but is still printed out with an empty list
- `x` is in conflict with `y` if ???
- is it worth to write a function that returns true if two trees are in conflict?

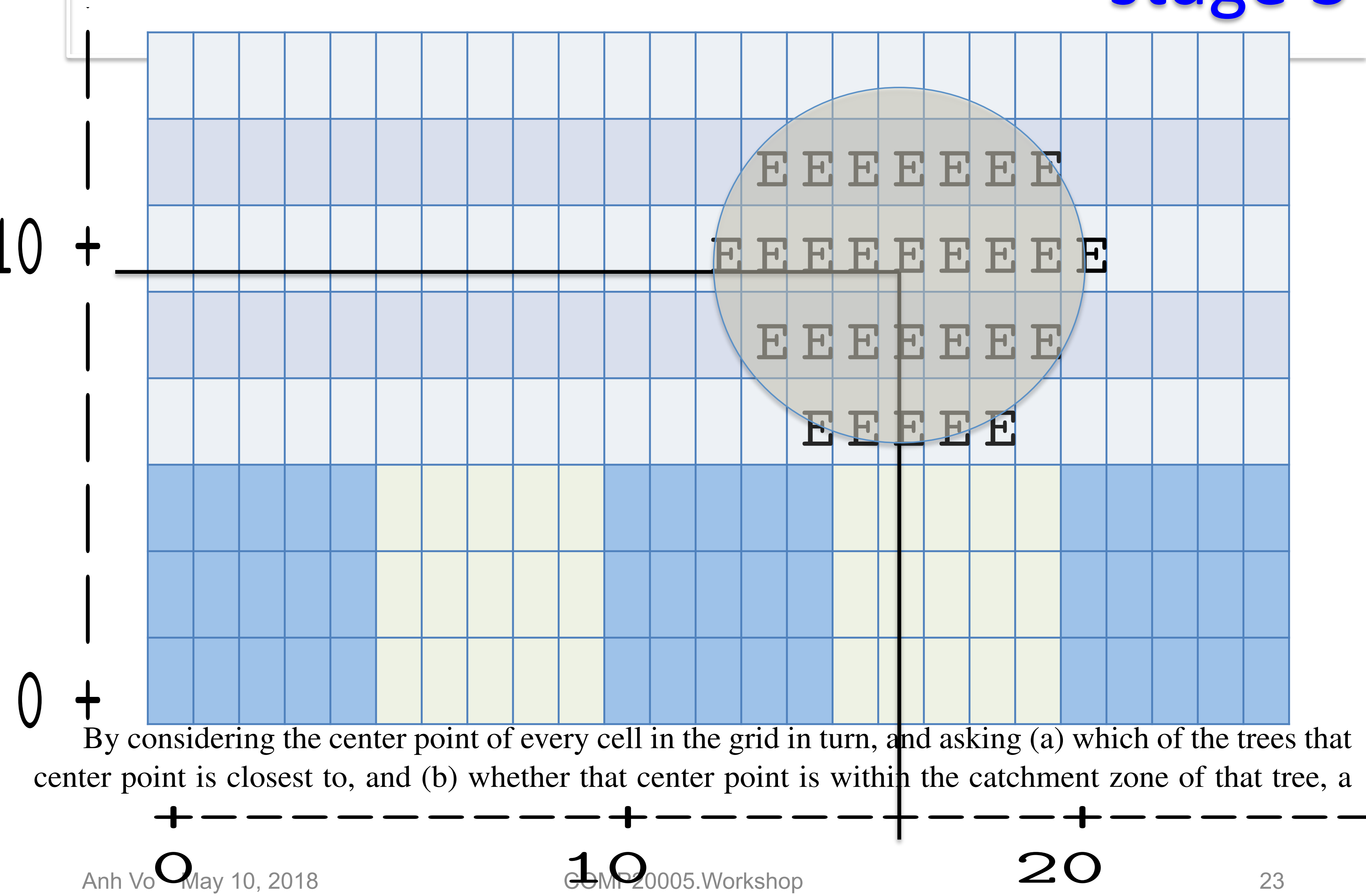
## stage 3

total 30 rows, 70 columns (or *vice versa*)
center of cell `[1][2]` is (2.5, 3.0)
center of cell `[i][j]` for row `i` column `j` is ?
Beware: here `i` represents `y`, `j` represents `x` …



```
E          16.5      10.5      150000   4.2
```

## stage 3



By considering the center point of every cell in the grid in turn, and asking (a) which of the trees that center point is closest to, and (b) whether that center point is within the catchment zone of that tree, a

## Tutor Quality of Teaching (QoT) survey

https://apps.eng.unimelb.edu.au/casmas/index.php?r=qoct/subjects

(link is likely provided in LMS)