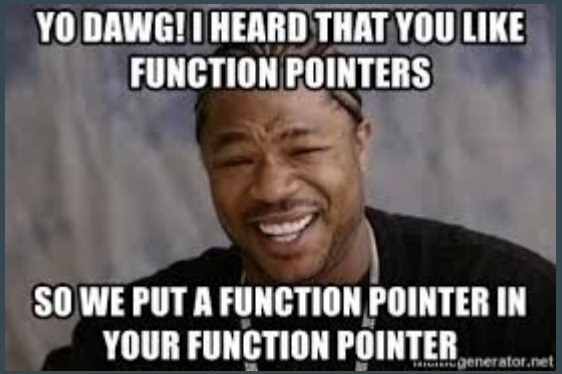
A few of our favourite things

...

Bash && (function) Pointers
2310 Study Exercises
Week 6, Sem 1, 2020



What this is and isn't

This slide deck **is not**:

- Not a comprehensive revision session
- Not a list of what will, or will not be examined (or examinable)
- Not necessarily the precise style and format of exam questions
- Not necessarily the same complexity or difficulty level as exam questions
- Not a substitute for your own study, revision, practice

This slide deck **is**:

- A good recap on some lecture content
- Is inspired from past exams and teaching-staff experience

Please familiarise yourself with the ECP, and be up-to-date with lectures.

Answers are in colour. If you disagree with an answer, post on piazza.

Bash & shell commands

Write shell commands to do the following:

1. List the unique users who are running commands on moss containing push2310 in reverse alphabetical order.
2. Count how many instances of vim each user on moss has open, excluding any with push2310 as a command line argument.

Bash & shell commands (possible answer)

Write shell commands to do the following:

1. List the unique users who are running commands on moss containing push2310 in reverse alphabetical order.

List all processes

Filter out unwanted ones

Extract the user ID

Sort in reverse order

Remove duplicates

```
ps -eF | grep "push2310" | cut -d ' ' -f 1 | sort -r | uniq
```

Can also use unique flag in sort. Alternative solutions also exist.

Bash & shell commands (possible answer)

Write shell commands to do the following:

2. Count how many instances of vim each user on moss has open, excluding any with push2310 as a command line argument.

List all processes

Find those containing vim

Filter the ones which don't contain push2310

Truncate consecutive spaces (to make cut work correctly)

Select the user ID

```
ps -eF | grep "vim" | grep -v 'push2310' | tr -s ' ' | cut -d ' '
-f 1 | sort | uniq -c
```

Sort the output

Count the number of instances

Pointers

Describe the following declarations in (plain) English.

```
1.  int** a;
2.  char** c[];
3.  double f[][];
4.  void (*g)();
5.  void* (*abcd)(void);
6.  void* (*def)(int*, char* a[]);
7.  char* (*(*foo)(const int*))[4];
8.  void* (*(*f)(int, char *(*)(void), float (*)()))(int);
9.  int (*bar)(void* (*)(int), void* (*(*)(int, char**, double)));
```

Pointers: 1

```
int **a;
```

Pointers: 1

a is a pointer to a pointer to an int.

```
int **a;
```

Pointers: 2

```
char** c[];
```

Pointers: 2

c is an array ...

```
char** c[];
```

Pointers: 2

c is an array of pointers to pointers to chars

```
char** c[];
```

Pointers: 3

```
double f[][];
```

Pointers: 3

f is an array ...

```
double f[][];
```

Pointers: 3

f is an array of arrays ...

```
double f[][]
```

Pointers: 3

f is an array of arrays of doubles

(i.e. f is a 2D array of doubles)

```
double f[][];
```

Pointers: 4

```
void (*g)();
```

Pointers: 4

g is pointer to a function ...

```
void (*g)();
```

Pointers: 4

g is pointer to a function which takes an unspecified number of arguments ...

```
void (*g)();
```

Pointers: 4

g is pointer to a function which takes an unspecified number of arguments and returns nothing (void)

```
void (*g)();
```

Pointers: 5

```
void* (*abcd)(void);
```

Pointers: 5

abcd is pointer to a function ...

```
void* (*abcd)(void);
```

Pointers: 5

abcd is pointer to a function which takes no parameters ...

```
void* (*abcd)(void);
```

Pointers: 5

abcd is pointer to a function which takes no parameters and returns a void pointer.

```
void* (*abcd)(void);
```

Pointers: 6

```
void* (*def)(int*, char* a[]);
```

Pointers: 6

def is a pointer to a function ...

```
void* (*def)(int*, char* a[]);
```

Pointers: 6

```
def is a pointer to a function which takes as
parameters: ...
```

```
void* (*def)(int*, char* a[]);
```

Pointers: 6

def is a pointer to a function which takes as parameters: an int pointer ...

```
void* (*def)(int*, char* a[]);
```

Pointers: 6

```
def is a pointer to a function which takes as
parameters: an int pointer and an array of
char* ...
```

```
void* (*def)(int*, char* a[]);
```

Pointers: 6

def is a pointer to a function which takes as parameters: an int pointer and an array of char* and returns a void pointer

```
void* (*def)(int*, char* a[]);
```

Pointers: 7

```
char* (*(*foo)(const int*))[4];
```

Pointers: 7

Foo is a pointer to a function ...

```
char* (*(*foo)(const int*))[4];
```

Pointers: 7

Foo is a pointer to a function which takes as parameters: a pointer to a constant int ...

```
char* (*(*foo)(const int*))[4];
```

Pointers: 7

Foo is a pointer to a function which takes as parameters: a pointer to a constant int and returns ...

```
char* (*(*foo)(const int*))[4]
```

Pointers: 7

Foo is a pointer to a function which takes as parameters: a pointer to a constant int and returns a pointer ...

```
char* (*(*foo)(const int*))[4]
```

Pointers: 7

Foo is a pointer to a function which takes as parameters: a pointer to a constant int and returns a pointer to an array of 4 ...

```
char* (*(*foo)(const int*))[4]
```

Pointers: 7

Foo is a pointer to a function which takes as parameters: a pointer to a constant int and returns a pointer to an array of 4 pointers to chars.

```
char* (*(*foo)(const int*))[4]
```

Pointers: 8

```
void* (*(*f)(int, char* (*)(void), float (*)()))(int);
```

Pointers: 8

f is a pointer to a function ...

```
void* (*(*f)(int, char* (*)(void), float (*)()))(int);
```

Pointers: 8

f is a pointer to a function which takes as parameters ...

```
void* (*(*f)(int, char* (*)(void), float (*)()))(int);
```

Pointers: 8

f is a pointer to a function which takes as parameters:

- an int ...

```
void* (*(*f)(int, char* (*)(void), float (*)()))(int);
```

Pointers: 8

f is a pointer to a function which takes as parameters:

- an int,
- a pointer to a function ...

```
void* (*(*f)(int, char* (*)(void), float (*)()))(int);
```

Pointers: 8

f is a pointer to a function which takes as parameters:

- an int,
- a pointer to a function which:
  - takes no parameters ...

```
void* (*(*f)(int, char* (*)(void), float (*)()))(int);
```

Pointers: 8

f is a pointer to a function which takes as parameters:

- an int,
- a pointer to a function which:
  - takes no parameters
  - returns a char pointer ...

```
void* (*(*f)(int, char* (*)(void), float (*)()))(int);
```

Pointers: 8

f is a pointer to a function which takes as parameters:

- an int,
- a pointer to a function which:
  - takes no parameters
  - returns a char pointer
- a pointer to a function ...

```
void* (*(*f)(int, char* (*)(void), float (*)()))(int);
```

Pointers: 8

f is a pointer to a function which takes as parameters:

- an int,
- a pointer to a function which:
  - takes no parameters
  - returns a char pointer
- a pointer to a function which:
  - takes an unspecified number of parameters ...

```
void* (*(*f)(int, char* (*)(void), float (*)()))(int);
```

Pointers: 8

f is a pointer to a function which takes as parameters:

- an int,
- a pointer to a function which:
  - takes no parameters
  - returns a char pointer
- a pointer to a function which:
  - takes an unspecified number of parameters
  - returns a float...

```
void* (*(*f)(int, char* (*)(void), float (*)()))(int);
```

Pointers: 8

f is a pointer to a function which takes as parameters:

- an int,
- a pointer to a function which:
  - takes no parameters
  - returns a char pointer
- a pointer to a function which:
  - takes an unspecified number of parameters
  - returns a float

and returns ...

```
void* (*(*f)(int, char* (*)(void), float (*)()))(int);
```

Pointers: 8

f is a pointer to a function which takes as parameters:

- an int,
- a pointer to a function which:
  - takes no parameters
  - returns a char pointer
- a pointer to a function which:
  - takes an unspecified number of parameters
  - returns a float

and returns a pointer to a function ...

```
void* (*(*f)(int, char* (*)(void), float (*)()))(int);
```

Pointers: 8

f is a pointer to a function which takes as parameters:

- an int,
- a pointer to a function which:
  - takes no parameters
  - returns a char pointer
- a pointer to a function which:
  - takes an unspecified number of parameters
  - returns a float

and returns a pointer to a function which takes as parameters: an int ...

```
void* (*(*f)(int, char* (*)(void), float (*)()))(int);
```

Pointers: 8

f is a pointer to a function which takes as parameters:

- an int,
- a pointer to a function which:
  - takes no parameters
  - returns a char pointer
- a pointer to a function which:
  - takes an unspecified number of parameters
  - returns a float

and returns a pointer to a function which takes as parameters: an int, and returns a void pointer

```
void* (*(*f)(int, char* (*)(void), float (*)()))(int);
```

Pointers: 9

```
int (*bar)(void* (*)(int), void* (*(*)(int, char**, double)));
```

Pointers: 9

bar is a pointer to a function ...

```
int (*bar)(void* (*)(int), void* (*(*)(int, char**, double)));
```

Pointers: 9

bar is a pointer to a function which takes as parameters ...

```
int (*bar)(void* (*)(int), void* (*(*)(int, char**, double)));
```

Pointers: 9

bar is a pointer to a function which takes as parameters:

- a pointer to a function ...

```
int (*bar)(void* (*)(int), void* (*(*)(int, char**, double)));
```

Pointers: 9

bar is a pointer to a function which takes as parameters:

- a pointer to a function which takes as parameters: an int ...

```
int (*bar)(void* (*)(int), void* (*(*)(int, char**, double)));
```

Pointers: 9

bar is a pointer to a function which takes as parameters:

- a pointer to a function which takes as parameters: an int, and returns a void pointer ...

```
int (*bar)(void* (*)(int), void* (*(*)(int, char**, double)));
```

Pointers: 9

bar is a pointer to a function which takes as parameters:

- a pointer to a function which takes as parameters: an int, and returns a void pointer
- a pointer to a function ...

```
int (*bar)(void* (*)(int), void* (*(*)(int, char**, double)));
```

Pointers: 9

bar is a pointer to a function which takes as parameters:

- a pointer to a function which takes as parameters: an int, and returns a void pointer
- a pointer to a function which takes as parameters ...

```
int (*bar)(void* (*)(int), void* (*(*)(int, char**, double)));
```

Pointers: 9

bar is a pointer to a function which takes as parameters:

- a pointer to a function which takes as parameters: an int, and returns a void pointer
- a pointer to a function which takes as parameters: an int ...

```
int (*bar)(void* (*)(int), void* (*(*)(int, char**, double)));
```

Pointers: 9

bar is a pointer to a function which takes as parameters:

- a pointer to a function which takes as parameters: an int, and returns a void pointer
- a pointer to a function which takes as parameters: an int, a pointer to a char pointer ...

```
int (*bar)(void* (*)(int), void* (*(*)(int, char**  double)));
```

Pointers: 9

bar is a pointer to a function which takes as parameters:

- a pointer to a function which takes as parameters: an int, and returns a void pointer
- a pointer to a function which takes as parameters: an int, a pointer to a char pointer, and a double ...

```
int (*bar)(void* (*)(int), void* (*(*)(int, char**, double)));
```

Pointers: 9

bar is a pointer to a function which takes as parameters:

- a pointer to a function which takes as parameters: an int, and returns a void pointer
- a pointer to a function which takes as parameters: an int, a pointer to a char pointer, and a double, and returns a pointer ...

```
int (*bar)(void* (*)(int), void* (*(*)(int, char**, double)));
```

Pointers: 9

bar is a pointer to a function which takes as parameters:

- a pointer to a function which takes as parameters: an int, and returns a void pointer
- a pointer to a function which takes as parameters: an int, a pointer to a char pointer, and a double, and returns a pointer to a void pointer ...

```
int (*bar)(void* (*)(int), void* (*(*)(int, char**, double)));
```

Pointers: 9

bar is a pointer to a function which takes as parameters:

- a pointer to a function which takes as parameters: an int, and returns a void pointer
- a pointer to a function which takes as parameters: an int, a pointer to a char pointer, and a double, and returns a pointer to a void pointer

and returns an int

```
int (*bar)(void* (*)(int), void* (*(*)(int, char**, double)));
```

Pointers

Describe the following declarations in (plain) English.

```
1.  int** a;
2.  char** c[];
3.  double f[][];
4.  void (*g)();
5.  void* (*abcd)(void);
6.  void* (*def)(int*, char* a[]);
7.  char* (*(*foo)(const int*))[4];
8.  void* (*(*f)(int, char* (*)(void), float (*)()))(int);
9.  int (*bar)(void* (*)(int), void* (*(*)(int, char**, double)));
```

Pointers (answers)

1. `int** a;`
   a is a pointer to a pointer to an int.
2. `char** c[];`
   c is an array of pointers to pointers to chars.
3. `double f[][];`
   f is a 2D array of doubles;
4. `void (*g)();`
   g is a pointer to a function that takes an unspecified number of parameters and returns nothing.
5. `void* (*abcd)(void);`
   abcd is a pointer to a function that takes no parameters and returns a void pointer.

Pointers (answers)

6. `void* (*def)(int*, char* a[]);`
   def is a pointer to a function that takes as parameters a pointer to an int and an array of pointers to chars, and returns a void pointer.
7. `char *(*(*foo)(const int *))[4];`
   foo is a pointer to a function that takes a pointer to a constant int, and returns a pointer to an array of 4 pointers to chars.
8. `void *(*(*f)(int, char *(*)(void), float (*)())(int));`
   f is a pointer to a function that takes as parameters:
   - int
   - pointer to a function which takes no parameters, and returns a char pointer
   - pointer to a function taking unspecified parameters and returning a float

   And returns a pointer to a function that takes a single int and returns a pointer to void.

Pointers (answers)

9. `int (*bar)(void* (*)(int), void* (*(*)(int, char**, double)));`

   bar is a pointer to a function that takes as parameters:
   - a pointer to a function which takes a single int and returns a void pointer
   - a pointer to a function which takes as parameters:
     - int
     - char double pointer
     - double

     and returns a pointer to a void pointer

   and returns an int.

2310 Week 7: Assignment 2 due @ **18:00 Tuesday**

- <u>Hard deadline for A2. No late attempts or submissions.</u>
- Don’t leave it too late.
  - Moss can get slow near the deadline