Static Code Analysis
Guest Lecture by Jun Yuan

Finding bad printf calls
- Trace where non-constant format string comes from
  - Use type checking to guard against format string bugs
    - Types are expressive
  - We can use untainted char * and tainted char *
    - untainted is a subtype of tainted (subtyping)
    - This allows the programmer to specify input which may come from the user
  - We can require printf to take an untainted char * as its first argument

```c
printf(untainted char * hello);
tainted char * read_from_user();
```

Subtyping on functions
\[ t_1' \subseteq t_1, t_2' \subseteq t_2 \]
\[ (t_1 \rightarrow t_2) \subseteq (t_1' \rightarrow t_2') \]

```c
tainted int * p;
untainted int * q;
p = q;
```

Ccured: safe typed pointers
- Check safe typed pointers and only insert checks for unsafe typed pointers

```c
int a[3];
int * p;
p = a + 4;
```

Give bounds to p, it can only go to a + 3

Jone & Kelly
- If p is never used this isn't an issue
- What if we decrement p before dereferencing it
- When a pointer goes out of bounds keep track of it (OOB Object)
  - Remove it when pointer goes back in bounds
- Fat pointers
  - Add necessary dynamic checks to fat pointers