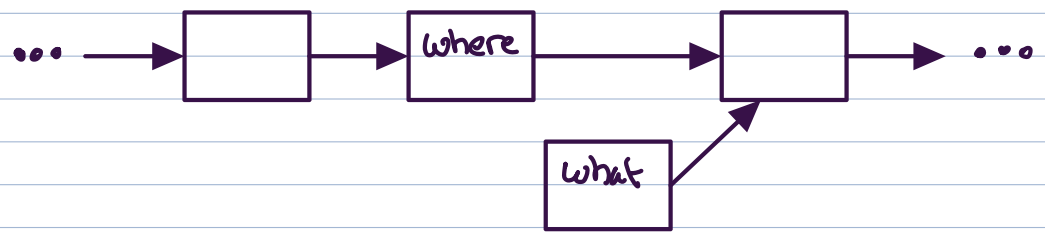
Husam Almanakly & Michael Bentivegna 12-6-21

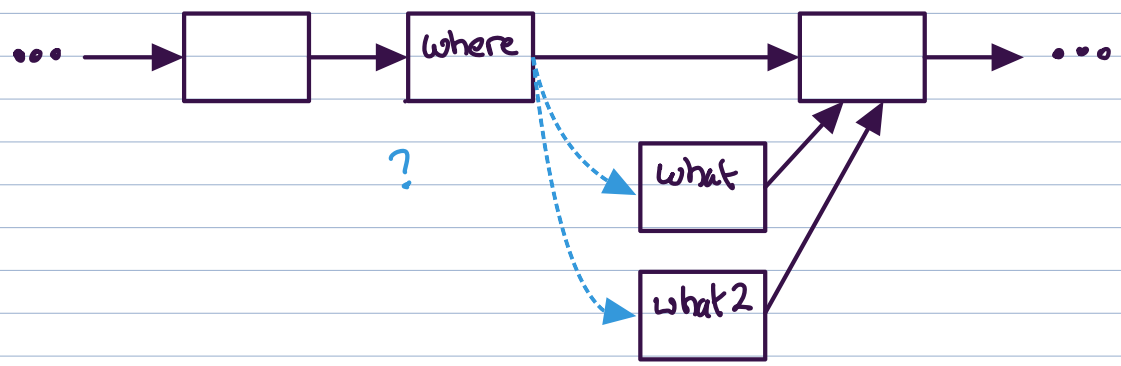
OS Problem Set 6

1) a) The given function is not safe in a concurrent situation because 2 processes could attempt to insert after the where node at the same time. Consider the initial linked list

Inserting what, after the first line we have

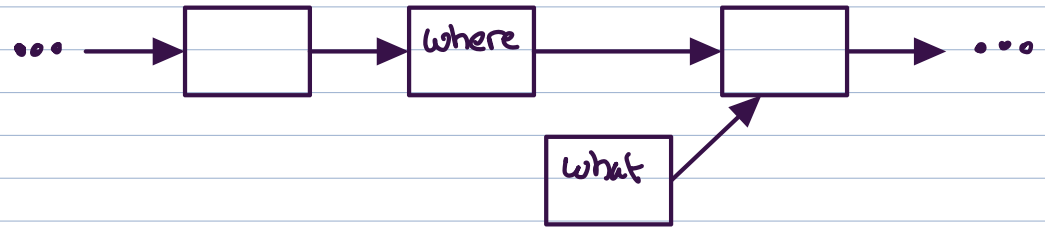


Assuming another thread tries to insert after where now results in ...



At this point, its actually indeterminate which struct the where node points to - it would update to one of the two whats

Similarly, for the STV case, at the same point in the insert (after the first line)



If a signal comes in & interrupts the process, the function could never 'finish'. Resulting in a dangling node

b) void ll_insert(struct ll * where, struct ll *what)

{

```
sigset_t oldmask, newmask;
    sigset_fill(&newmask);
    sigprocmask(SIG_BLOCK, &newmask, &oldmask);
what -> fwd = where -> fwd;
where -> fwd = what;
    sigprocmask(SIG_SETMASK, &oldmask, NULL);
```

}

c) void ll_insert(struct ll * where, struct ll *what)

{

```
while (TAS(&where -> spinlock))
    ;
what -> fwd = where -> fwd;
where -> fwd = what;
where -> spinlock = 0
```

}