CSE 142
Computer Programming I

Event Driven Programming

K-1

Event-Driven Programming

Modern programs tend to be "event-driven"

Program starts, sets itself up.

Program enters a loop, waiting for some event or command to happen:

mouse click, key click, timer, menu selection, etc.

Program performs operation ("handles" the event or command)

Program goes back to its wait loop

UW's GP142 graphics package follows this kind of model

K-2

Simple Command Interpreter

Repeatedly read in "commands" and handle them.

Input (symbolized by single characters)

a -- execute command A by calling *process_A( )*

b -- execute command B by calling *process_B( )*

q -- quit

Pseudocode for main loop:

get next command

if a, execute command A

if b, execute command B

if q, signal quit

K-3

Command Interpreter
Loop Control Schema

repeat until quit signal

use variable "done" to indicate when done

```
set done to false
while not done {
    body statements
    if quit command, set done to true
}
```

K-4

**Command
Interpreter
main ( )**

```c
int main(void) {
    char command;
    int done;

    done = FALSE;
    while (! done){     /* Input command from user */
        command = ReadCommand( );
        switch (command){
        case 'A':
        case 'a':
            process_A( );       /* Execute command A */
            break;
        case 'B':
        case 'b':
            process_B( );       /* Execute command B */
            break;
        case 'Q':
        case 'q':
            done = TRUE;      /* quit */
            break;
        default:
            printf("Unrecognized command\n");
        }
    }
    return 0;
}
```

K-5