Switch

Last updated 12/17/20

Switch

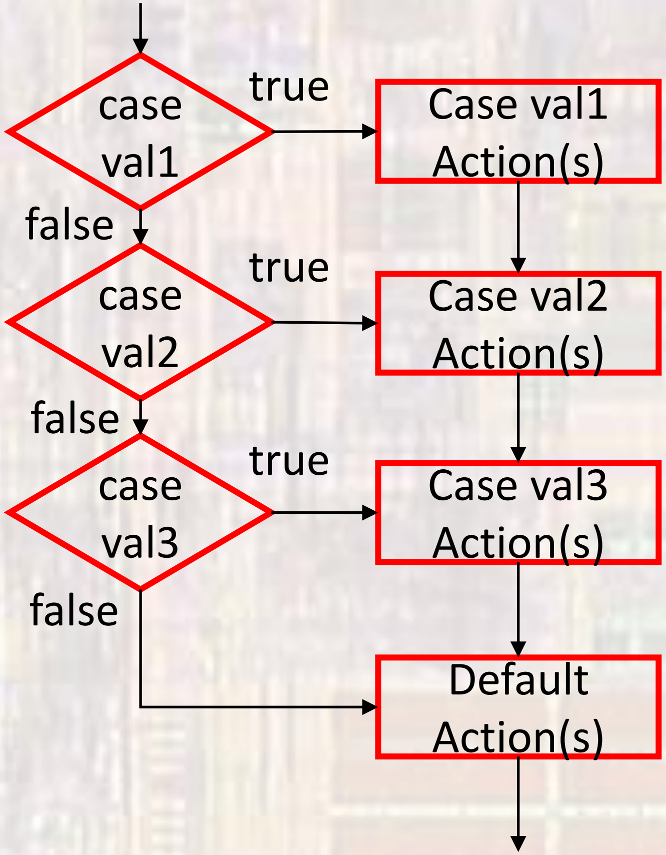
- These slides introduce the switch statement
- Upon completion: You should be able to interpret and code solutions using the switch statement

Switch

- Switch Statement
  - If… else allows a 2 way decision
  - Switch allows for n-way decisions
  - More restrictive on the tests

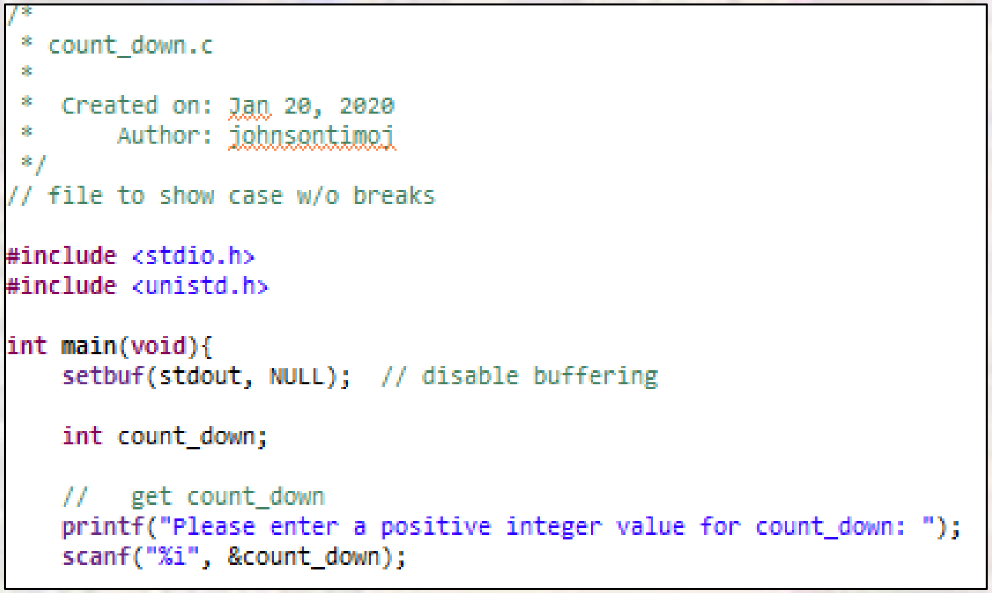
```
…
switch(variable){
  case val1: statement;
  case val2: statement;
  case val3: statement;
  default: statement;
}
```

variable must be an integral value

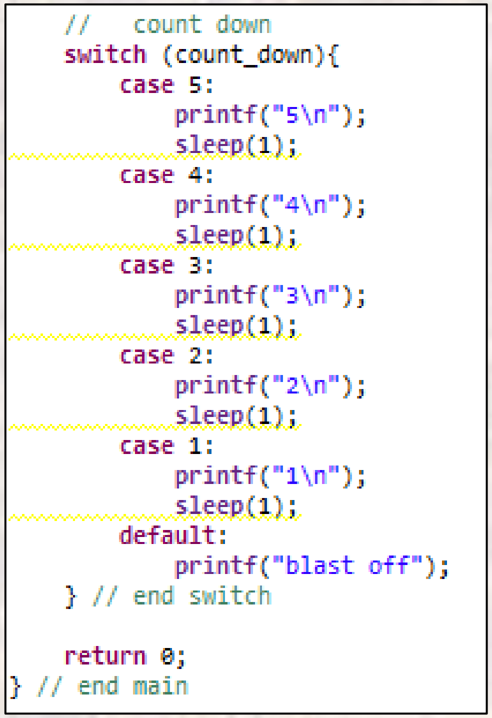


Switch

- Switch Statement
  - Launch Countdown



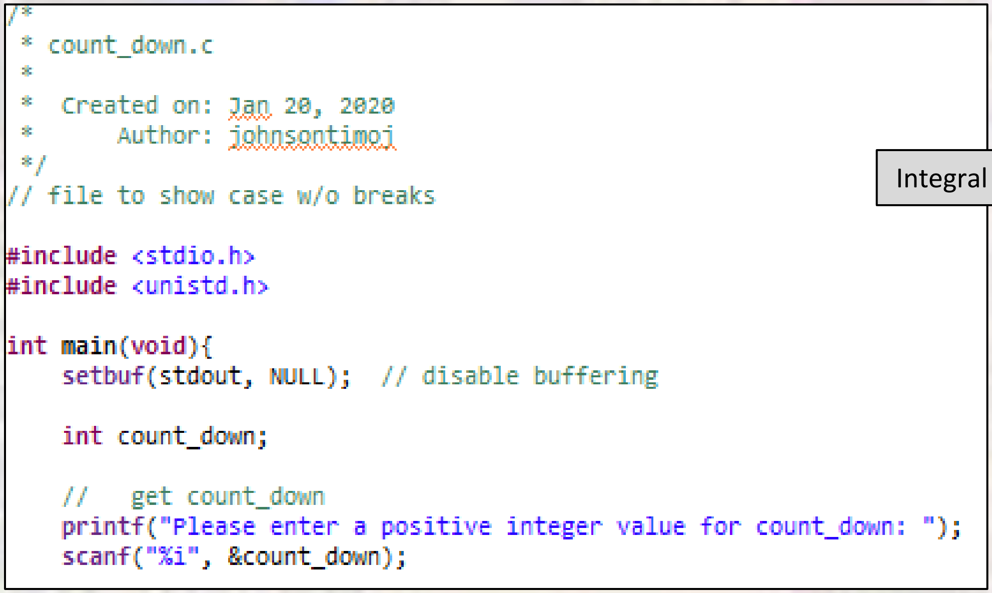
```
/*
 * count_down.c
 *
 *  Created on: Jan 20, 2020
 *      Author: johnsontimoj
 */
// file to show case w/o breaks

#include <stdio.h>
#include <unistd.h>

int main(void){
    setbuf(stdout, NULL);  // disable buffering

    int count_down;

    //  get count_down
    printf("Please enter a positive integer value for count_down: ");
    scanf("%i", &count_down);
```



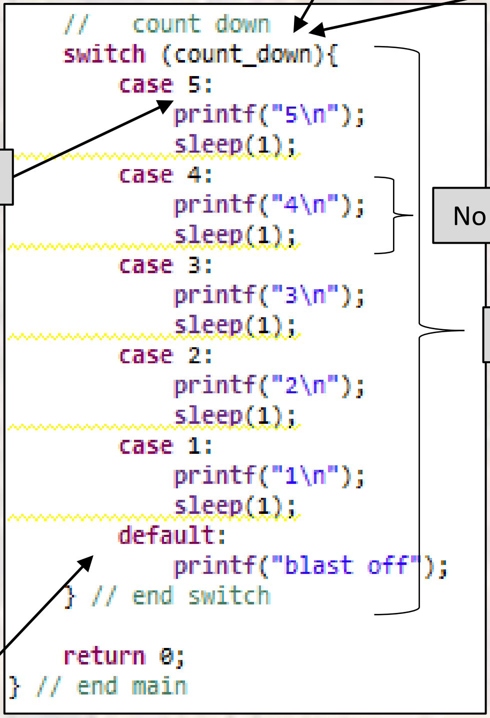
```
    //  count down
    switch (count_down){
        case 5:
            printf("5\n");
            sleep(1);
        case 4:
            printf("4\n");
            sleep(1);
        case 3:
            printf("3\n");
            sleep(1);
        case 2:
            printf("2\n");
            sleep(1);
        case 1:
            printf("1\n");
            sleep(1);
        default:
            printf("blast off");
    } // end switch

    return 0;
} // end main
```

Switch

- Switch Statement
  - Launch Countdown



```
/*
 * count_down.c
 *
 *  Created on: Jan 20, 2020
 *      Author: johnsontimoj
 */
// file to show case w/o breaks

#include <stdio.h>
#include <unistd.h>

int main(void){
    setbuf(stdout, NULL);  // disable buffering

    int count_down;

    //   get count_down
    printf("Please enter a positive integer value for count_down: ");
    scanf("%i", &count_down);
```



```
    //   count down
    switch (count_down){
        case 5:
            printf("5\n");
            sleep(1);
        case 4:
            printf("4\n");
            sleep(1);
        case 3:
            printf("3\n");
            sleep(1);
        case 2:
            printf("2\n");
            sleep(1);
        case 1:
            printf("1\n");
            sleep(1);
        default:
            printf("blast off");
    } // end switch

    return 0;
} // end main
```

variable to test

Cannot be an expression

Integral value

No { … } required

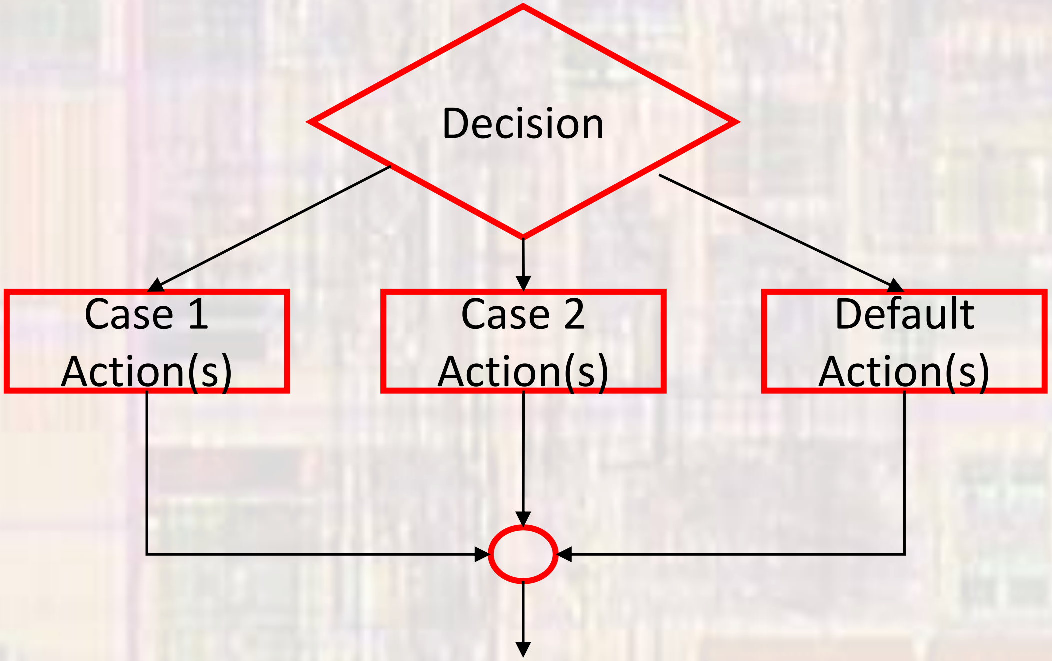
{ … } required

Always include a default even if empty

Switch

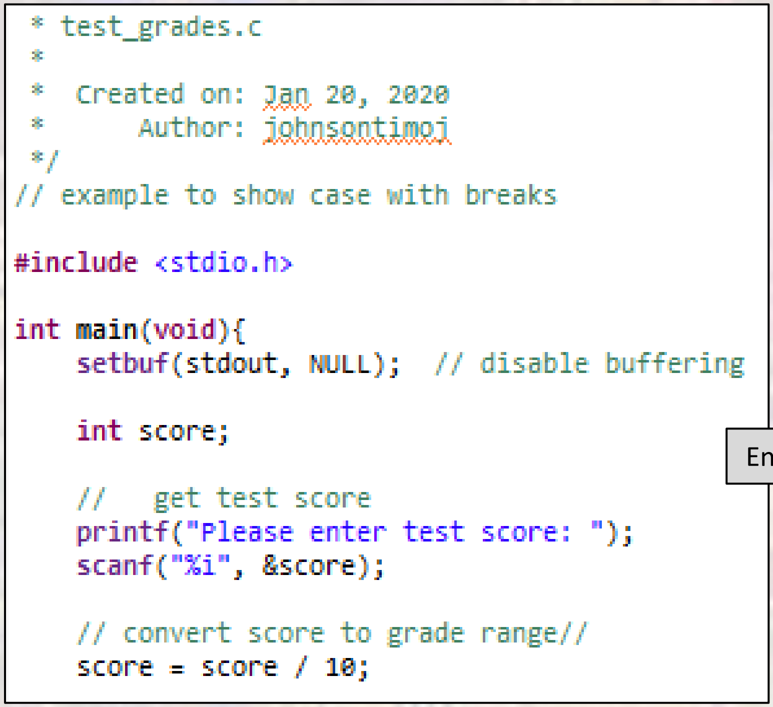
- Switch Statement
  - Switch with break
    - Break statement: causes the current case to terminate and the switch to complete

```
switch(variable){
 case val1:      statement;
                 statement;
                 break;
 case val1:      statement;
                 statement;
                 break;
 default:        statement;
                 statement;
                 break;
}
```

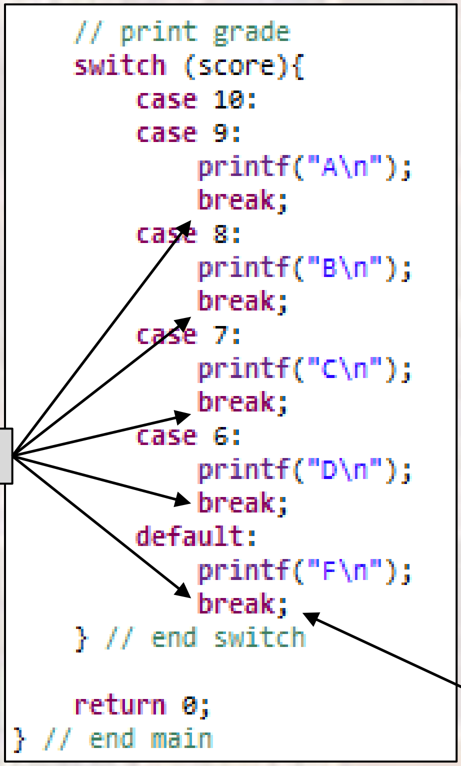


Switch

- Switch Statement
  - Switch with break



```
 * test_grades.c
 *
 *  Created on: Jan 20, 2020
 *      Author: johnsontimoj
 */
// example to show case with breaks

#include <stdio.h>

int main(void){
    setbuf(stdout, NULL);   // disable buffering

    int score;

    //   get test score
    printf("Please enter test score: ");
    scanf("%i", &score);

    // convert score to grade range//
    score = score / 10;
```



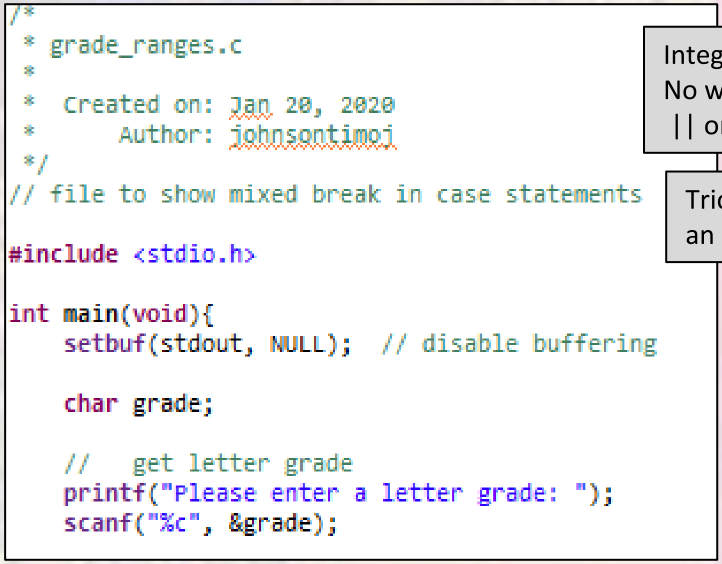
```
    // print grade
    switch (score){
        case 10:
        case 9:
            printf("A\n");
            break;
        case 8:
            printf("B\n");
            break;
        case 7:
            printf("C\n");
            break;
        case 6:
            printf("D\n");
            break;
        default:
            printf("F\n");
            break;
    } // end switch

    return 0;
} // end main
```

End switch here

Not necessary but considered a best practice

Switch

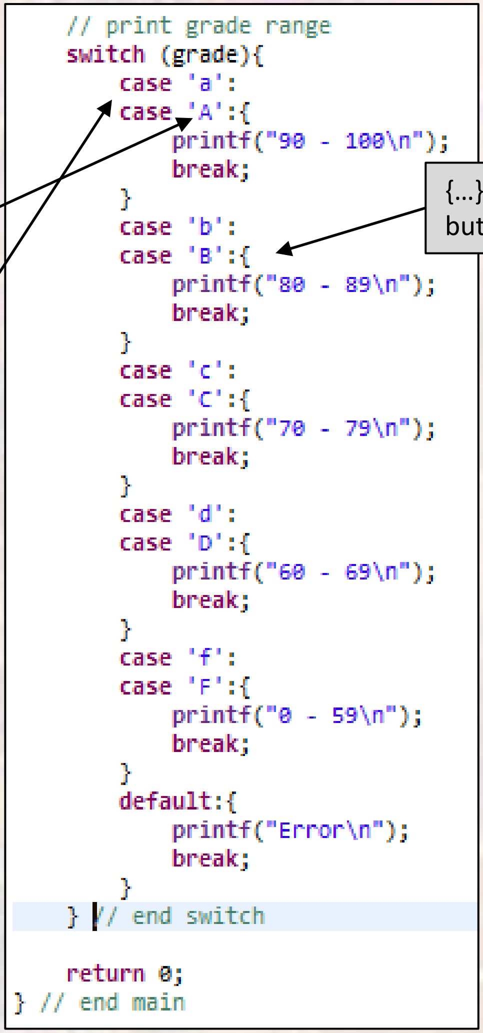
- Switch Statement
  - Mixing break and fall-through



```
/*
 * grade_ranges.c
 *
 *  Created on: Jan 20, 2020
 *      Author: johnsontimoj
 */
// file to show mixed break in case statements

#include <stdio.h>

int main(void){
    setbuf(stdout, NULL);  // disable buffering

    char grade;

    //  get letter grade
    printf("Please enter a letter grade: ");
    scanf("%c", &grade);
```

Integral value -
No way to do
|| or &&

Trick to emulate
an OR

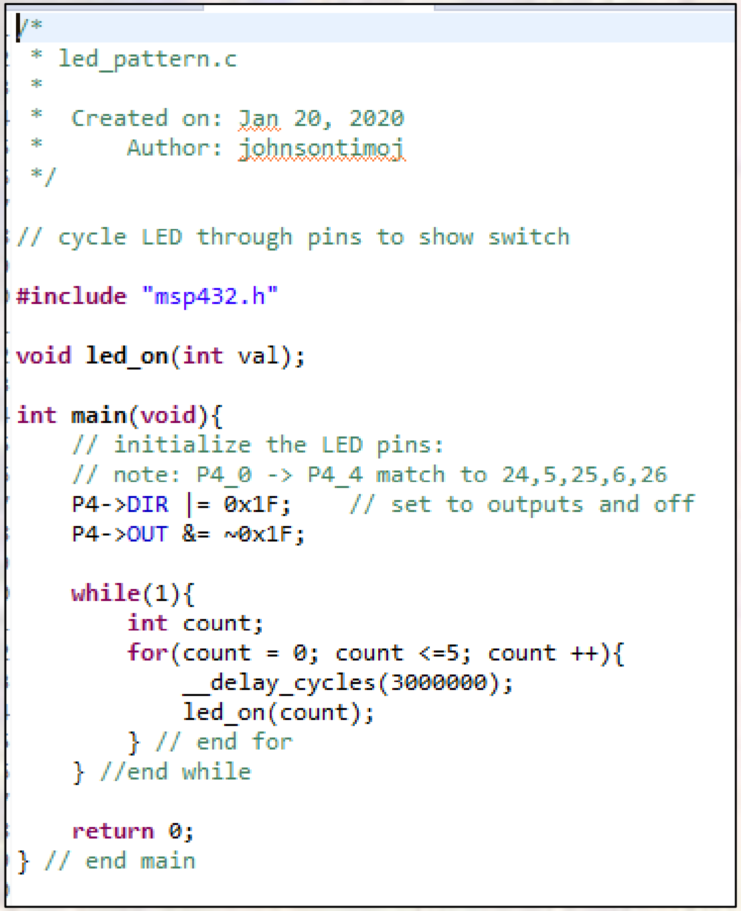


```
    // print grade range
    switch (grade){
        case 'a':
        case 'A':{
            printf("90 - 100\n");
            break;
        }
        case 'b':
        case 'B':{
            printf("80 - 89\n");
            break;
        }
        case 'c':
        case 'C':{
            printf("70 - 79\n");
            break;
        }
        case 'd':
        case 'D':{
            printf("60 - 69\n");
            break;
        }
        case 'f':
        case 'F':{
            printf("0 - 59\n");
            break;
        }
        default:{
            printf("Error\n");
            break;
        }
    } // end switch

    return 0;
} // end main
```

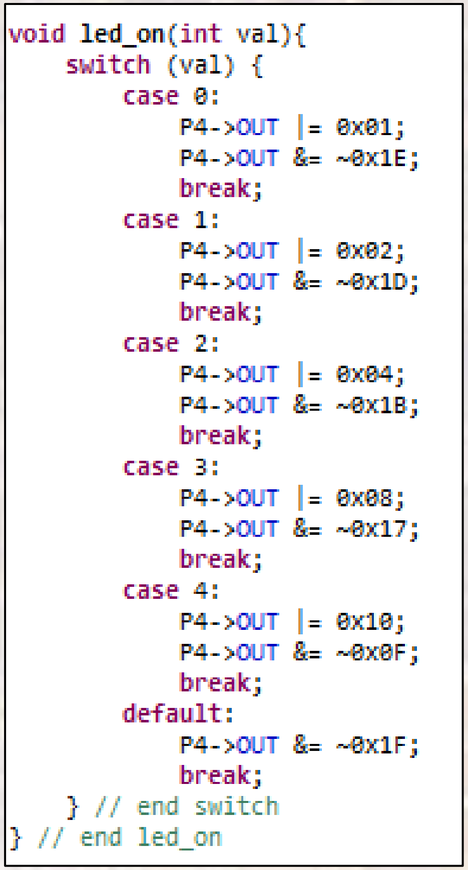
{…} not required
but allowed for clarity

Switch

- Turn on LEDs in a pattern
    - p4_0 -> p4_5 = 24,5,25,6,26



```
/*
 * led_pattern.c
 *
 *  Created on: Jan 20, 2020
 *      Author: johnsontimoj
 */

// cycle LED through pins to show switch

#include "msp432.h"

void led_on(int val);

int main(void){
    // initialize the LED pins:
    // note: P4_0 -> P4_4 match to 24,5,25,6,26
    P4->DIR |= 0x1F;     // set to outputs and off
    P4->OUT &= ~0x1F;

    while(1){
        int count;
        for(count = 0; count <=5; count ++){
            __delay_cycles(3000000);
            led_on(count);
        } // end for
    } //end while

    return 0;
} // end main
```



```
void led_on(int val){
    switch (val) {
        case 0:
            P4->OUT |= 0x01;
            P4->OUT &= ~0x1E;
            break;
        case 1:
            P4->OUT |= 0x02;
            P4->OUT &= ~0x1D;
            break;
        case 2:
            P4->OUT |= 0x04;
            P4->OUT &= ~0x1B;
            break;
        case 3:
            P4->OUT |= 0x08;
            P4->OUT &= ~0x17;
            break;
        case 4:
            P4->OUT |= 0x10;
            P4->OUT &= ~0x0F;
            break;
        default:
            P4->OUT &= ~0x1F;
            break;
    } // end switch
} // end led_on
```