1. LED example

DDRB	Description
0	Input pin
1	Output pin

PORTB	Description
0	Output low value
1	Output high value

DDRB	PORTB	Direction	Internal pull-up resistor	Description
0	0	Input	No	Tri-state, high-impedance
0	1	Input	Yes	PORTB will source current if ext. pulled low.
1	0	Output	No	Output low (Sink)
1	1	Output	No	Output high (Source)

Port	Pin	Input/output usage?
A	x	Microcontroller ATmega328P does not contain port A
B	0	Yes (Arduino pin 8)
	1	Yes (Arduino pin ~9)
	2	Yes (Arduino pin ~10)
	3	Yes (Arduino pin ~11)
	4	Yes (Arduino pin 12)
	5	Yes (Arduino pin 13)
	6	No (xtal clock generator)
	7	No (xtal clock generator)
C	0	Yes (Arduino pin A0)
	1	Yes (Arduino pin A1)
	2	Yes (Arduino pin A2)
	3	Yes (Arduino pin A3)
	4	Yes (Arduino pin A4)
	5	Yes (Arduino pin A5)
	6	No
	7	No
D	0	Yes (Arduino pin RX<-0)
	1	Yes (Arduino pin TX->0)
	2	Yes (Arduino pin 2)
	3	Yes (Arduino pin ~3)
	4	Yes (Arduino pin 4)
	5	Yes (Arduino pin ~5)
	6	Yes (Arduino pin ~6)
	7	Yes (Arduino pin 7)

LED example C code:

```c
/***********************************************************************
 *
 * Alternately toggle two LEDs when a push button is pressed.
 * ATmega328P (Arduino Uno), 16 MHz, AVR 8-bit Toolchain 3.6.2
 *
 * Copyright (c) 2018-2020 Tomas Fryza
 * Dept. of Radio Electronics, Brno University of Technology, Czechia
 * This work is licensed under the terms of the MIT license.
 *
 **********************************************************************/

/* Defines -----------------------------------------------------------*/
#define LED_GREEN   PB5     // AVR pin where green LED is connected
#define LED_RED     PC0
#define BTN                PD0
#define BLINK_DELAY 250
#ifndef F_CPU
#define F_CPU 16000000      // CPU frequency in Hz required for delay
#endif

/* Includes ----------------------------------------------------------*/
#include <util/delay.h>     // Functions for busy-wait delay loops
#include <avr/io.h>         // AVR device-specific IO definitions

/* Functions ---------------------------------------------------------*/
/**
 * Main function where the program execution begins. Toggle two LEDs
 * when a push button is pressed.
 */
int main(void)
{
    /* GREEN LED */
    // Set pin as output in Data Direction Register...
    DDRB = DDRB | (1<<LED_GREEN);
    // ...and turn LED off in Data Register
    PORTB = PORTB & ~(1<<LED_GREEN);

    /* second LED */
       DDRC = DDRC | (1<<LED_RED);
       PORTC = PORTC & ~(1<<LED_RED);

       /* button */
       DDRD = DDRD & ~(1<<LED_GREEN);
       PORTD = PORTD | (1<<BTN);

    // Infinite loop
    while (1)
    {
             _delay_ms(BLINK_DELAY);

             loop_until_bit_is_clear(PIND, BTN);

             PORTB = PORTB ^ (1<<LED_GREEN);
             PORTC = PORTC ^ (1<<LED_RED);

    }

    // Will never reach this
    return 0;
}
```

62. KNIGHT RIDER application:

C code:

```
/***********************************************************************
 *
 * Alternately toggle two LEDs when a push button is pressed.
 * ATmega328P (Arduino Uno), 16 MHz, AVR 8-bit Toolchain 3.6.2
 *
 * Copyright (c) 2018-2020 Tomas Fryza
 * Dept. of Radio Electronics, Brno University of Technology, Czechia
 * This work is licensed under the terms of the MIT license.
 *
 **********************************************************************/

/* Defines -----------------------------------------------------------*/
#define LED_1       PB1
#define LED_2       PB2
#define LED_3       PB3
#define LED_4       PB4
#define LED_5       PB5
#define BTN                PD0
#define BLINK_DELAY 200
#ifndef F_CPU
#define F_CPU 16000000      // CPU frequency in Hz required for delay
#endif

/* Includes ----------------------------------------------------------*/
#include <util/delay.h>     // Functions for busy-wait delay loops
#include <avr/io.h>         // AVR device-specific IO definitions

/* Functions ---------------------------------------------------------*/
/**
 * Main function where the program execution begins. Toggle two LEDs
 * when a push button is pressed.
 */
int main(void)
{
    /* INITIALIZATION */
    /* LED 1 */
       // Set pin as output in Data Direction Register...
    DDRB = DDRB | (1<<LED_1);
    // ...and turn LED off in Data Register
    PORTB = PORTB & ~(1<<LED_1);

    /* LED 2 */
       DDRB = DDRB | (1<<LED_2);
       PORTB = PORTB & ~(1<<LED_2);

    /* LED 3 */
    DDRB = DDRB | (1<<LED_3);
    PORTB = PORTB & ~(1<<LED_3);

    /* LED 4 */
    DDRB = DDRB | (1<<LED_4);
    PORTB = PORTB & ~(1<<LED_4);

    /* LED 5 */
    DDRB = DDRB | (1<<LED_5);
    PORTB = PORTB & ~(1<<LED_5);
```

```
        /* button */
        DDRD = DDRD & ~(1<<BTN);
        PORTD = PORTD | (1<<BTN);

     PORTB = PORTB ^ (1<<LED_1);                              // turn LED1 on
        _delay_ms(BLINK_DELAY);                                       // wait

        // Infinite loop
     while (1)
     {
               if (bit_is_clear(PIND, BTN))
               {
               /* FORWARD */
                      PORTB = PORTB ^ (1<<LED_1);             // turn LED1 off
                      PORTB = PORTB ^ (1<<LED_2);             // turn LED2 on
                      _delay_ms(BLINK_DELAY);                         // wait 200 ms

                      PORTB = PORTB ^ (1<<LED_2);             // turn LED2 off
                      PORTB = PORTB ^ (1<<LED_3);             // turn LED3 on
                      _delay_ms(BLINK_DELAY);                         // wait 200 ms

                      PORTB = PORTB ^ (1<<LED_3);             // turn LED3 off
                      PORTB = PORTB ^ (1<<LED_4);             // turn LED4 on
                      _delay_ms(BLINK_DELAY);                         // wait 200 ms

                      PORTB = PORTB ^ (1<<LED_4);             // turn LED4 off
                      PORTB = PORTB ^ (1<<LED_5);             // turn LED5 on
                      _delay_ms(BLINK_DELAY);                         // wait 200 ms

               /* BACK */
                      PORTB = PORTB ^ (1<<LED_5);             // turn LED5 off
                      PORTB = PORTB ^ (1<<LED_4);             // turn LED4 on
                      _delay_ms(BLINK_DELAY);                         // wait 200 ms

                      PORTB = PORTB ^ (1<<LED_4);             // turn LED4 off
                      PORTB = PORTB ^ (1<<LED_3);             // turn LED3 on
                      _delay_ms(BLINK_DELAY);                         // wait 200 ms

                      PORTB = PORTB ^ (1<<LED_3);             // turn LED3 off
                      PORTB = PORTB ^ (1<<LED_2);             // turn LED2 on
                      _delay_ms(BLINK_DELAY);                         // wait 200 ms

                      PORTB = PORTB ^ (1<<LED_2);             // turn LED2 off
                      PORTB = PORTB ^ (1<<LED_1);             // turn LED1 on
                      _delay_ms(BLINK_DELAY);                         // wait 200 ms
               }

     }

     // Will never reach this
     return 0;
}
```