PRZESUNIĘCIE BITOWE

w lewo <<
w prawo >>

[variable] << [numbersOfPlaces]

a	a<<1	a<<2	a>>1
0001	0010	0100	0000
0011	0110	1100	0001
0101	1010	0100	0010

```
#include <stdio.h>
int main()
{
    int a = 6;
    printf("6 <<2 = %d \n", a<<2);/* wypisze 24 */
    printf("6 >>2 = %d \n", a>>2);/* wypisze 1 */
```

PRZESUNIĘCIE BITOWE ARYTMETYCZNE w PRAWO
to powiela najstarszy bit

LISTA 1

ZAD 1

```
uint32_t wyzeruj_bit(uint32_t x, uint32_t k){
    return x & (~(1<<k));}
```

ustaw, czyli zapal

```
uint32_t ustaw_bit(
    return x|(1<<k);}

uint32_t zaneguj_bit(
    return x^(1<<k);}
```

ZAD. 2.

```
#include <stdint.h>
#include <stdio.h>
```

$x \cdot 2^y$

```
uint32_t p_1(uint32_t x, uint32_t y){
    return x<<y;}
```

$\lfloor x/2^y \rfloor$

```
uint32_t p_2(
    return x>>y;}
```

$x \bmod 2^y$

```
uint32_t p_3(
    return x&((1<<y)-1);}
```

z przesunięciem $x \& ((x << (32-y)) >> (32-y))$

$\lceil x/2^y \rceil$

```
uint32_t p_4(
    return (x>>y+((x>>(y-1))&1));}
```

ZAD 3.

$x = x+y$
$y = x-y$
$x = x-y$

ZA POMOCĄ XOR

xor	1	0
1	0	1
0	1	0

```
#include <stdio.h>
int main()
{
    int a=4, b=7
```

$a = a \wedge b$
$b = b \wedge a$
$a = a \wedge b$

ZAD 4.

MOJE

A) 1001

-8 +1

1111 1001

-128+64+32+16+8 +1

$-2^{x+4} + (2^{x+3} + 2^{x+2} + 2^{x+1} + 2^{x})$

$2^{x+4} - 2^{x}$

-2^{x}

TABLICA

$-2^{w+k} + \sum_{i=w}^{w+k-p} 2^{i} = -2^{w}$

$2^{w}(-2^{k} + 1 + 2 + 2^{2} + \dots + 2^{k-1}) = -1 \cdot 2^{w}$

$1 + 2 + 2^{2} + \dots + 2^{k-1} = -1 + 2^{k}$

$1 + 2^{k} - 2 = -1 + 2^{k}$

B) mogę obciąć w bitach jeśli w+1 są identyczne

ZAD 5. $(((x >> 1) - (\sim x \,\&\, 1)) >> 31) \,|\, (x \,\&\, (x-1))$

*ZAD 6.

```
uint32_t the_bit = ((x >> i) & 1) << k);
x = (x & ~(1 << k);
return x | the_bit
```

ZAD. 7. Ucita programistów str. 86

$y_1 = (x \,\&\, 0x55555555) + ((x \,\&\, xAAAAAAAA) >> 1)$

$y_2 = (y_1 \,\&\, 0x33333333) + ((x \,\&\, x0xCCCCCCCC) >> 2)$

$y_3 = (y_2 \,\&\, 0x0FF0FF0F) + ((x \,\&\, x0xF0F0F0F0) >> 4)$

$y_4 = (y_3 \,\&\, 0x00FF00FF) + ((x \,\&\, 0xFF00FF00) >> 8)$

$y_5 = (y_4 \,\&\, x0000FFFF) + ((x \,\&\, 0xFFFF0000) >> 16)$

return y_5;

ZAD. 8.

A B	C D
D C	B A

CD AB $x = (x >> 16) \,|\, (x << 16)$

DC BA $x = ((x \,\&\, x00FF00FF) << 8) \,|\, ((x \,\&\, xFF00FF00) >> 8)$

ZAD. 9

MOJE

A) $i \geq 0$

zmiana:

$i > 0$

$a[i-1] = a[i]$

B) $i - DELTA > 0$

zmiana:

$i \geq DELTA$

ZAD. 10

kod sterujący - kod, który w danym kodowaniu znaków nie przenosi informacji o samym znaku, ale służy do sterowania urządzeniem przetwarzającym dane, np. drukarka, terminalem

0 - Null
4 - End of Transmission (EOT)
7 - Bell (BEL)
10 - Line Feed (LF)
12 - Form Feed (FF)

ZAD. 11. Jakie ograniczenia standardu ASCII przyczyniły się do powstania UTF-8?

ASCII ma znaki amerykańskie, więc wykorzystywanie go w językach europejskich jest niemożliwe, gdyż w ASCII nie ma znaków diakrytycznych (np. ą, ż). Powstał unicode i jeden ze sposobów czytania go to UTF-8.

P	R	O	S	Ż	Ę	SPACJA	2
50	72	6f	73	7a	C4 99	20	7a

A	P	Ł	A	C	I	Ć
61	70	C5 82	61	63	69	C4 87

SPACJA	5	€	!
20	35	E2 82 AC	21