

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

Факультет прикладної математики
Кафедра програмного забезпечення комп'ютерних систем

ЗВІТ
з лабораторної роботи № 8
«Бітова інформація. Робота з бітовими операціями.
121 – Інженерія програмного забезпечення. 2021–2022 н.р.»

Дисципліна «Основи програмування»

Виконала :

студентка 1-го курсу, групи КП-11,
спеціальності 121 – Інженерія
програмного забезпечення
Кирильчук Олександра Артурівна

Перевірив:

к. т. н, старший викладач
Хайдуrow Владислав
Володимирович

Київ – 2021

Мета. Опанування основних операцій з бітами, закріплення знань з основних систем числення та представлення різних чисел у загальновживаних системах числення. Уміти розробляти найпростіші програми для роботи з бітовими операціями, а також закріпити набуті знання щодо пріоритетів операцій, які використовуються у мові програмування C#.

Оглавление

Завдання1	3
Завдання2	5

Завдання 1

Позначення (у мові програмування C#): “&” – побітове «і»; “|” – побітове «або»; “^” – побітове додавання за модулем 2 («XOR»).

12	$z \& y \& x \wedge y \wedge x x x z y \& x$
----	--

$z \& y \& x \wedge y \wedge x | x | x | z | y \& x$

$z = 125$
 $x = 68$ $y = 114$

I	II	III	IV	V	VI	VII	VIII	IX	X	XI	XII
x	y	z	x & y	x & z	y & z	x ^ y	x ^ z	y ^ z	x y	x z	y z
1	1	1	1	1	1	0	1	1	1	1	1
0	1	1	0	1	0	1	1	1	1	1	1
0	1	1	0	1	0	1	1	1	1	1	1
0	0	1	0	0	0	0	0	0	0	1	1
1	0	1	0	0	0	0	1	1	1	1	1
0	1	0	0	0	0	1	1	1	1	1	1
0	0	1	0	0	0	0	0	0	0	1	1
68	114	125	64	112	64	50	118	118	118	127	124

Signovige : 124 (1111111)

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Numerics;

namespace ConsoleApp12
{
    class Program
    {
        static void Main(string[] args)
        {
            Random rnd = new Random();
            int x, y, z;
            string X, Y, Z;
            bool b = false;
            do
            {
                x = rnd.Next(50, 150);
                y = rnd.Next(50, 150);
                z = rnd.Next(50, 150);
                X = Convert.ToString(x, 2);
                Y = Convert.ToString(y, 2);
            }
        }
    }
}
```

```

        Z = Convert.ToString(z, 2);
        if ((x - y) * (z - y) * (z - x) == 0) b = false;
        else b = true;
    }
    while (b == false);
    Console.WriteLine(X + " " + x);
    Console.WriteLine(Y + " " + y);
    Console.WriteLine(Z + " " + z);
    Solving(x, y, z);
}
static void Solving(int x, int y, int z)
{
    int t2 = y & x; Console.WriteLine("y & x = " + t2 + " " +
Convert.ToString(t2, 2));
    int t = z & y; Console.WriteLine("z & y = " + t + " " +
Convert.ToString(t, 2));
    int t1 = t & x; Console.WriteLine("z & y & x = " + t1 + " " +
Convert.ToString(t1, 2));
    int t3 = t1 ^ y; Console.WriteLine("z & y & x ^ y = " + t3 + " " +
Convert.ToString(t3, 2));
    int t4 = t3 ^ x; Console.WriteLine("z & y & x ^ y ^ x = " + t4 + " " +
Convert.ToString(t4, 2));
    int t5 = t4 | x; Console.WriteLine("z & y & x ^ y ^ x | x = " + t5 + " " +
Convert.ToString(t5, 2));
    int t6 = t5 | z; Console.WriteLine("z & y & x ^ y ^ x | x | z = " + t6 + " " +
+ Convert.ToString(t6, 2));
    int t7 = t6 | y; Console.WriteLine("z & y & x ^ y ^ x | x | z | y = " + t7 +
" " + Convert.ToString(t7, 2));
    int t8 = t7 | t2; Console.WriteLine("z & y & x ^ y ^ x | x | z | y & x =
" + t8 + " " + Convert.ToString(t8, 2));
    }
}
}

```

```

1000100 68
1110010 114
1111101 125
y & x = 64 1000000
z & y = 112 1110000
z & y & x = 64 1000000
z & y & x ^ y = 50 110010
z & y & x ^ y ^ x = 118 1110110
z & y & x ^ y ^ x | x = 118 1110110
z & y & x ^ y ^ x | x | x = 118 1110110
z & y & x ^ y ^ x | x | x | z = 127 1111111
z & y & x ^ y ^ x | x | x | z | y & x = 127 1111111

```

Завдання2

Шифр Вернама

Шифр Вернама (1926 р.). Відкритий текст кодується двійковою послідовністю, ключ та шифртекст також представляються послідовностями «0» і «1» такої ж довжини. ШТ отримується з ВТ і ключа операцією XOR.

ВТ	0	1	1	1	1	0	1	1	0	1	1	0	1	0	0
К	1	1	0	0	0	0	0	0	0	1	1	1	0	0	0
ШТ	1	0	1	1	1	0	1	1	0	0	0	1	1	0	0

Процедура шифрування

ШТ	1	0	1	1	1	0	1	1	0	0	0	1	1	0	0
К	1	1	0	0	0	0	0	0	0	1	1	1	0	0	0
ВТ	0	1	1	1	1	0	1	1	0	1	1	0	1	0	0

Процедура розшифрування

У якості ключа береться «ідеально» випадкова послідовність – послідовність незалежних рівномірних випадкових біт, тобто кожна реалізація довжини n з'являється з ймовірністю 2^n незалежно від ВТ. У даному прикладі ймовірність появи будь-якої ключової послідовності, а також будь-якої криптограми 1215. Даний шифр застосовується дотепер (так звана стрічка одноразового користування або одноразовий блокнот) на окремих важливих напрямках зв'язку. За допомогою побудованої теорії Шеннон довів, що цей шифр є цілком таємним, тобто маючи тільки криптограму ніяким чином неможливо знайти відкритий текст і навіть будь-яку інформацію щодо нього. Головним недоліком шифру Вернама є велика довжина ключа, що треба попередньо передавати по закритому каналу.

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Numerics;

namespace ConsoleApp12
{
    class Program
    {
        static void Main(string[] args)
        {
            Console.WriteLine("Input str : "); string str = Console.ReadLine();
            Console.WriteLine("Input key : "); string key = Console.ReadLine();
            string key1 = Key1(key, str);
            Verman(str, key1);
        }
        static string Key1(string key, string str)
        {
            char Key = Convert.ToChar(key[0]);
            char Str = Convert.ToChar(str[0]);
            for (int i = 0; i < str.Length; i++)
            {
                if(str.Length > key.Length)
```

```

        {
            key += Str % Key;
        }
    }
    return key;
}
static void Verman(string Str, string Key)
{
    char[] str = Str.ToCharArray();
    char[] key = Key.ToCharArray();
    char[] entext = Encryption(key, str);
    ReEncryption(key, entext);
}
static char[] Encryption(char[] key, char[] str)
{
    char[] entext = new char[str.Length];
    for (int i = 0; i < str.Length; i++)
    {
        entext[i] = (char)(str[i] ^ key[i]);
        Console.Write(entext[i]);
    }
    Console.WriteLine();
    return entext;
}
static void ReEncryption(char[] key, char[] entext)
{
    char[] reentext = new char[entext.Length];
    for (int i = 0; i < entext.Length; i++)
    {
        reentext[i] = (char)(entext[i] ^ key[i]);
        Console.Write(reentext[i]);
    }
    Console.WriteLine();
}
}
}

```

```

Input str :
readwantsleepreadytogo
Input key :
key
↓↑S@VYCD[RRGERVSNCPX
readwantsleepreadytogo

```