**00009620**

Task 1

<https://github.com/00009620/CSF->

Task 2

a) Convert my ID(9620) into binary code:

|  |  |  |
| --- | --- | --- |
| Division by 2 | Quotient | Remainder |
| 9620/2 | 4810 | 0 |
| 4810/2 | 2405 | 0 |
| 2405/2 | 1202 | 1 |
| 1202/2 | 601 | 0 |
| 601/2 | 300 | 1 |
| 300/2 | 150 | 0 |
| 150/2 | 75 | 0 |
| 75/2 | 37 | 1 |
| 37/2 | 18 | 1 |
| 18/2 | 9 | 0 |
| 9/2 | 4 | 1 |
| 4/2 | 2 | 0 |
| 2/2 | 1 | 0 |
| 1 | 0 | 1 |

Convert 9620 to hexadecimal

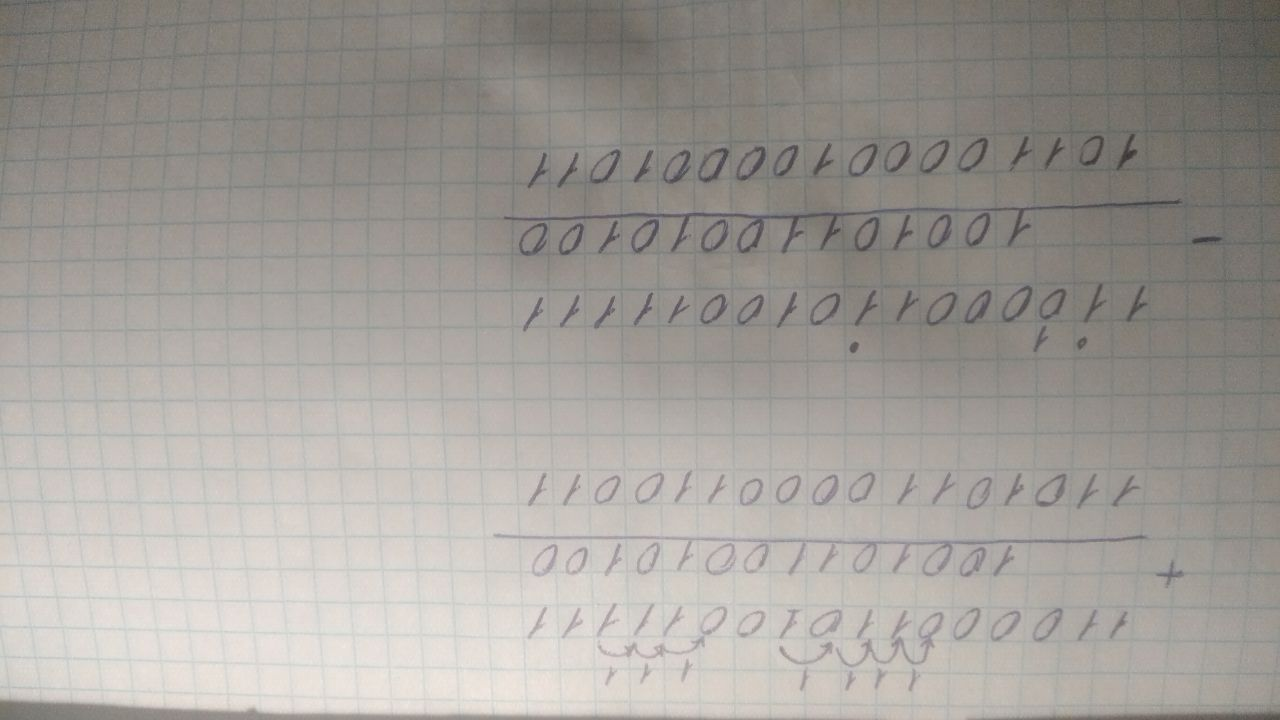
|  |  |  |
| --- | --- | --- |
| Division by 16 | Quotient | Remainder |
| 9620/16 | 601 | 4 |
| 601/16 | 37 | 9 |
| 37/16 | 2 | 5 |
| 2 | 0 | 2 |

Reverse the remainders and get answers.

Answer: 9620 in binary = 10010110010100

9620 in hex = 2594

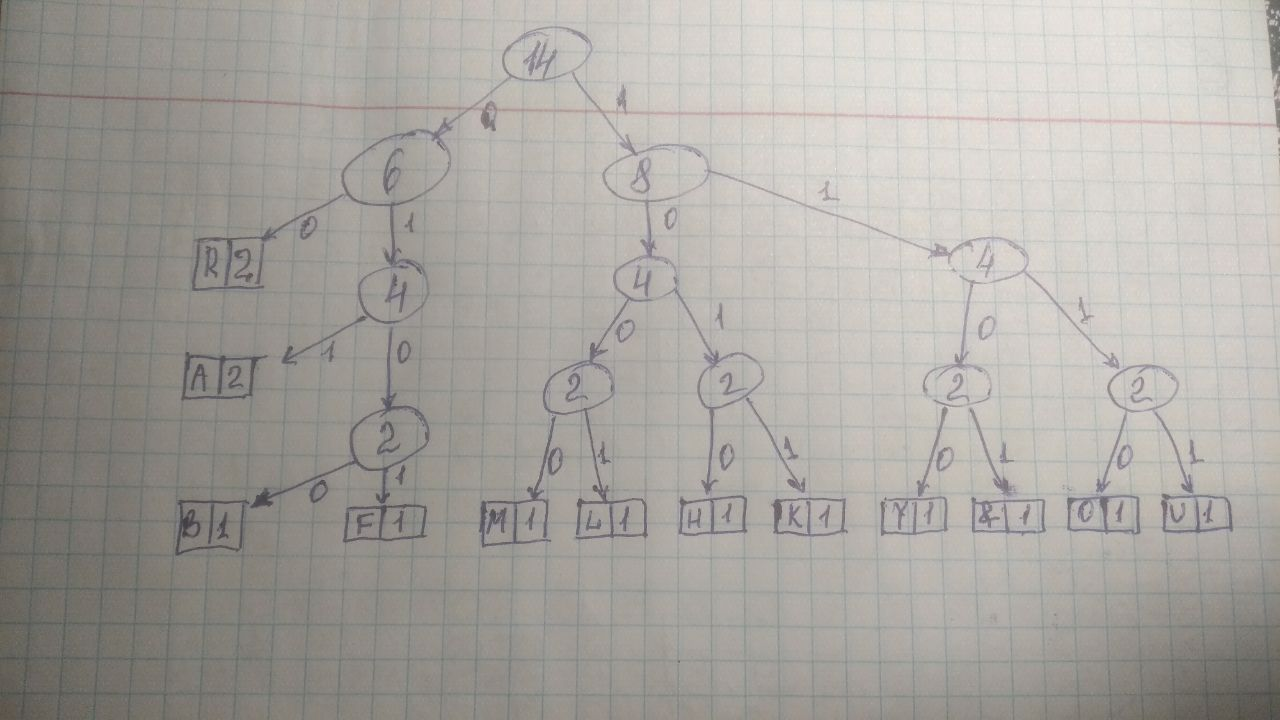
b) 99999 is equal to 11000011010011111 in binary code.

Below are the results of subtraction and addition :

c) Hexadecimal code is most used for space exemption or simply is can hold more numbers of a binary code, using just a couple of digits. So, it is really great in terms of space allocation. The only possible drawback is that it is a bit hard for people to understand it and convert it straightforward to decimal in a head.

Task 3

Let’s depict my parents’ names (BAKHROM&FLYURA) in Huffman’s tree.



Creating table to depict frequency and encoding for each letter:

|  |  |  |
| --- | --- | --- |
| Letters | Frequency | Encoding |
| R | 2 | 00 |
| A | 2 | 011 |
| B | 1 | 0100 |
| F | 1 | 0101 |
| M | 1 | 1000 |
| L | 1 | 1001 |
| H | 1 | 1010 |
| K | 1 | 1011 |
| Y | 1 | 1100 |
| & | 1 | 1101 |
| O | 1 | 1110 |
| U | 1 | 1111 |

Let’s now calculate number of bits that needed to be allocated to store this data:

2\*2+2\*3+1\*4+1\*4+1\*4+1\*4+1\*4+1\*4+1\*4+1\*4+1\*4+1\*4=50bits

Answer: 50bits