Zestaw zadań numer 3 do wykonania:

- nie ma terminu, nie ma punktów
- nie podsyłamy na żaden mail (ewentualnie można podesłać żebym sprawdził czy jest OK)
- na kolejne zajęcia można przynieść program.
- będzie kartkówka
- można korzystać z programu

Zadanie:

Na repozytorium znajduje się projekt **Binary**:
https://github.com/mpenarprz/InformatykaA1/tree/master/Sample/Binary

W środku pliku .cpp znajduje się definicja klasy EncodedNumber – stanowi ona naszą klasę bazową dla różnych „koderów" liczb binarnych. Można zapoznać się z jej definicją, ale tak naprawdę najbardziej istotne są dwa fakty:

- zdefiniowana jest metoda wirtualna: **virtual unsigned int encode(int originalValue, unsigned int absoluteVal),** która odpowiada za konwersję liczby na odpowiedni ciąg bitów. unsigned int traktujemy jako niezinterpretowany ciąg bitów. Wirtualność metody oznacza że implementacja jest w klasie pochodnej.
- zawiera metodę **print()** – pokazująca jaka jest wartość liczby w formacie o podstawie 8, 10, 16 i binarnym

Zadanie polega na dopisaniu do istniejących klas: **U2, ZM** oraz **BIAS** odpowiednich operacji w **encode()** które wyznaczają odpowiedni ciąg bitów.

Dla przykładu dorzuciłem 2 implementacje: U1 oraz NBC.

U1 implementuje metodę encode() jako:

```
unsigned int encode(int originalValue,  unsigned int absoluteVal) {
        return originalValue > 0 ? absoluteVal : ~absoluteVal;
}
```

NBC z kolei:

```
unsigned int encode(int originalValue, unsigned int absoluteVal) {
        if (originalValue >= 0) {
                return originalValue;
        }
        else {
                std::cout << "! Nie da rady !";
                return 0;
        }
}
```

Jeszcze raz:

Klasy U2/ZM i BIAS już są zdefiniowane. main() także. Zadanie sprowadza się do uzupełnienia 3 fragmentów:

```
unsigned int encode(int originalValue, unsigned int absoluteVal) {
	std::cout << "Nie zaimplementowano" << std::endl;
	return 0;
}
```