Sprawozdanie lab 5 – modulacja ciągła – Dominik Ciesiołkiewicz 44289 222B

Zad1:

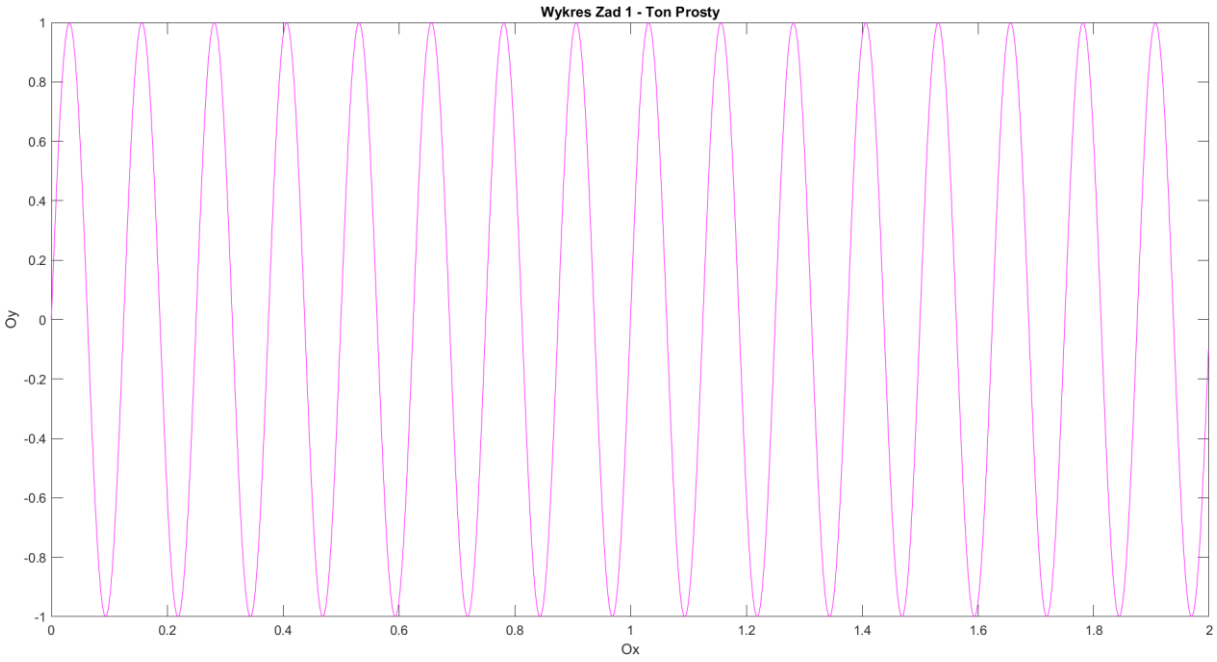
Kod:

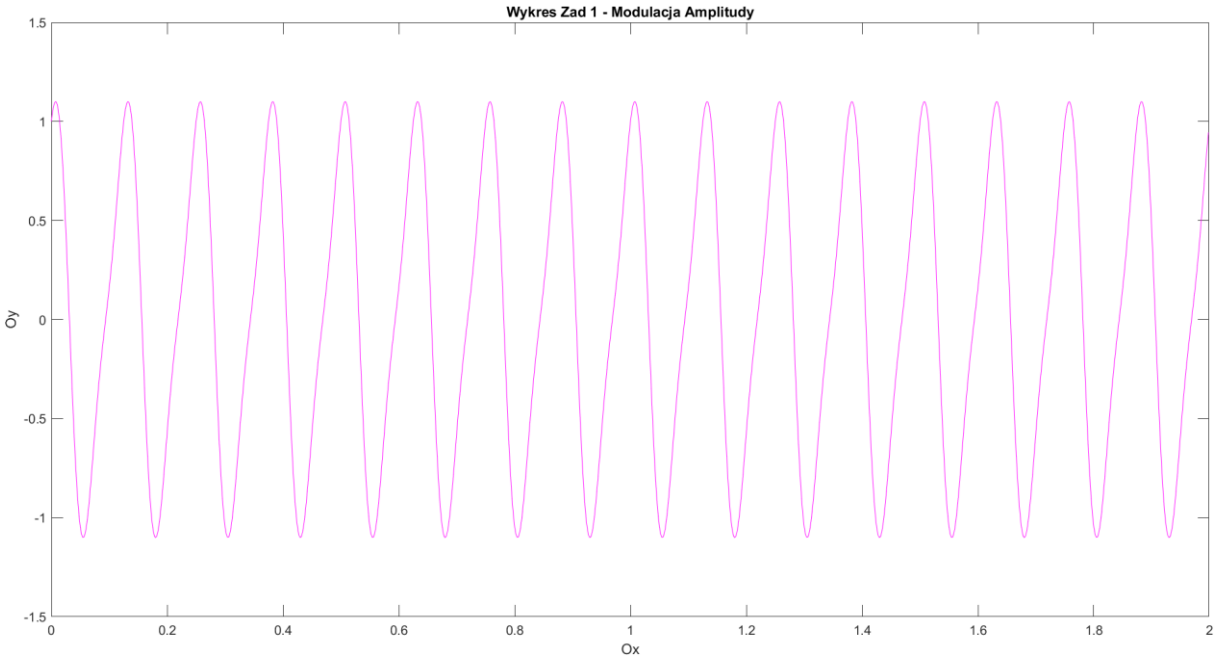
```
#include <iostream>
#include <complex>
#include <fstream>

#define _USE_MATH_DEFINES

double pi = 3.14159265359;

using namespace std;

double ton_prosty(double a, double F, double phi, double t)
{
	double s = a * sin(2 * pi * F * t + phi);
	return s;
}

int main()
{
	double a = 1;//volty
	double A = 2;//z numeru albumu
	double F = 8;
	double phi = 2 * pi;
	double fs = 500;// (?)
	double Ts = 1 / fs;

	double kA = 0.5;
	double kp = 1;

	ofstream saveOX("zad1OX.txt");
	ofstream saveTonProsty("zad1sig.txt");
	ofstream saveM("zad1M.txt");
	ofstream saveZa("zad1Za.txt");
	ofstream saveZp("zad1Zp.txt");

	int count = 0;

	for (double i = 0; i < A; i = i + Ts)
```

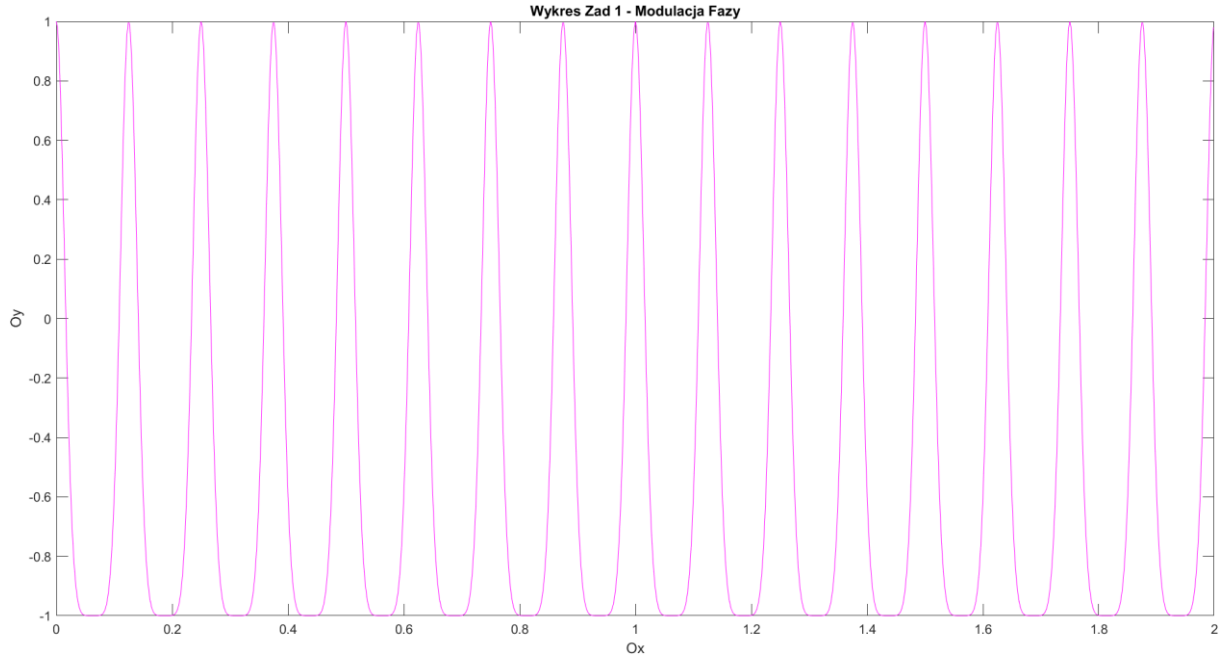
```
	{
		count++;
	}

	double* sig = new double[count];
	int ilosc = count;
	count = 0;

	double fn = F;

	for (double i = 0; i < A; i = i + Ts)
	{
		sig[count] = ton_prosty(a, F, phi, i);
		saveOX << i << endl;
		saveTonProsty << sig[count] << endl;

		double m = a * sin(2 * pi * F * i);
		saveM << m << endl;
		double Za = (kA * m + 1) * cos(2 * pi * fn * i);
		saveZa << Za << endl;
		double Zp = cos(2 * pi * fn * i + kp * m);
		saveZp << Zp << endl;

		count++;
	}

	saveOX.close();
	saveTonProsty.close();
	saveM.close();
	saveZa.close();
	saveZp.close();

	return 0;
}
```

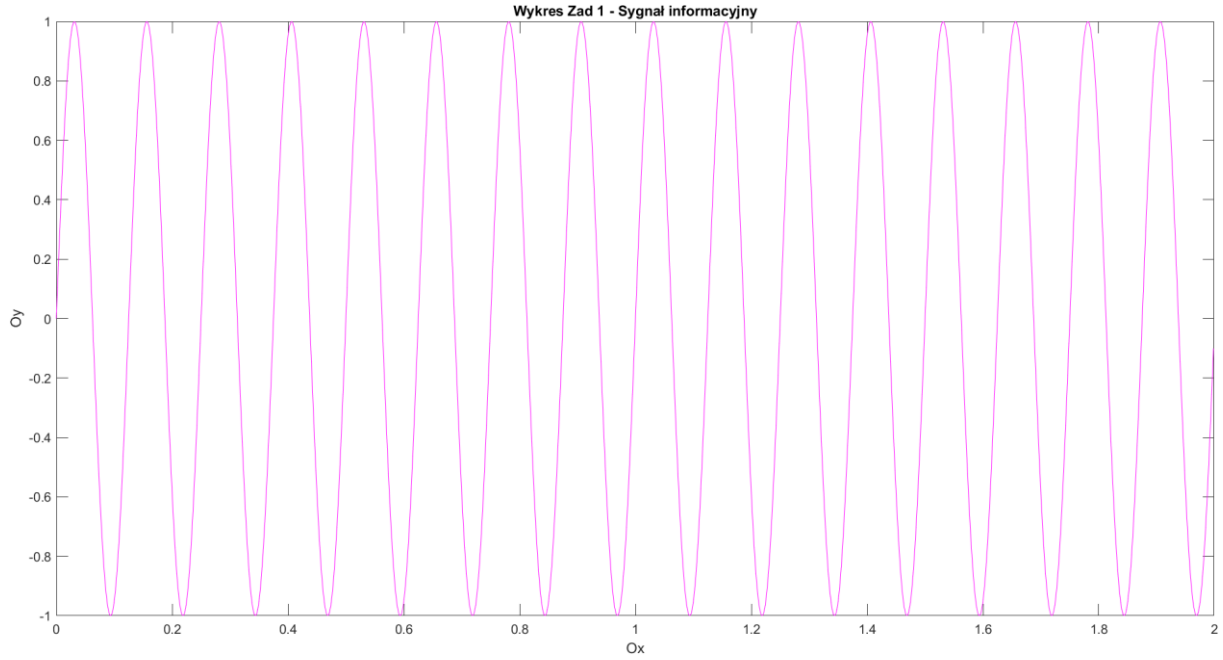
Wykresy:

a)

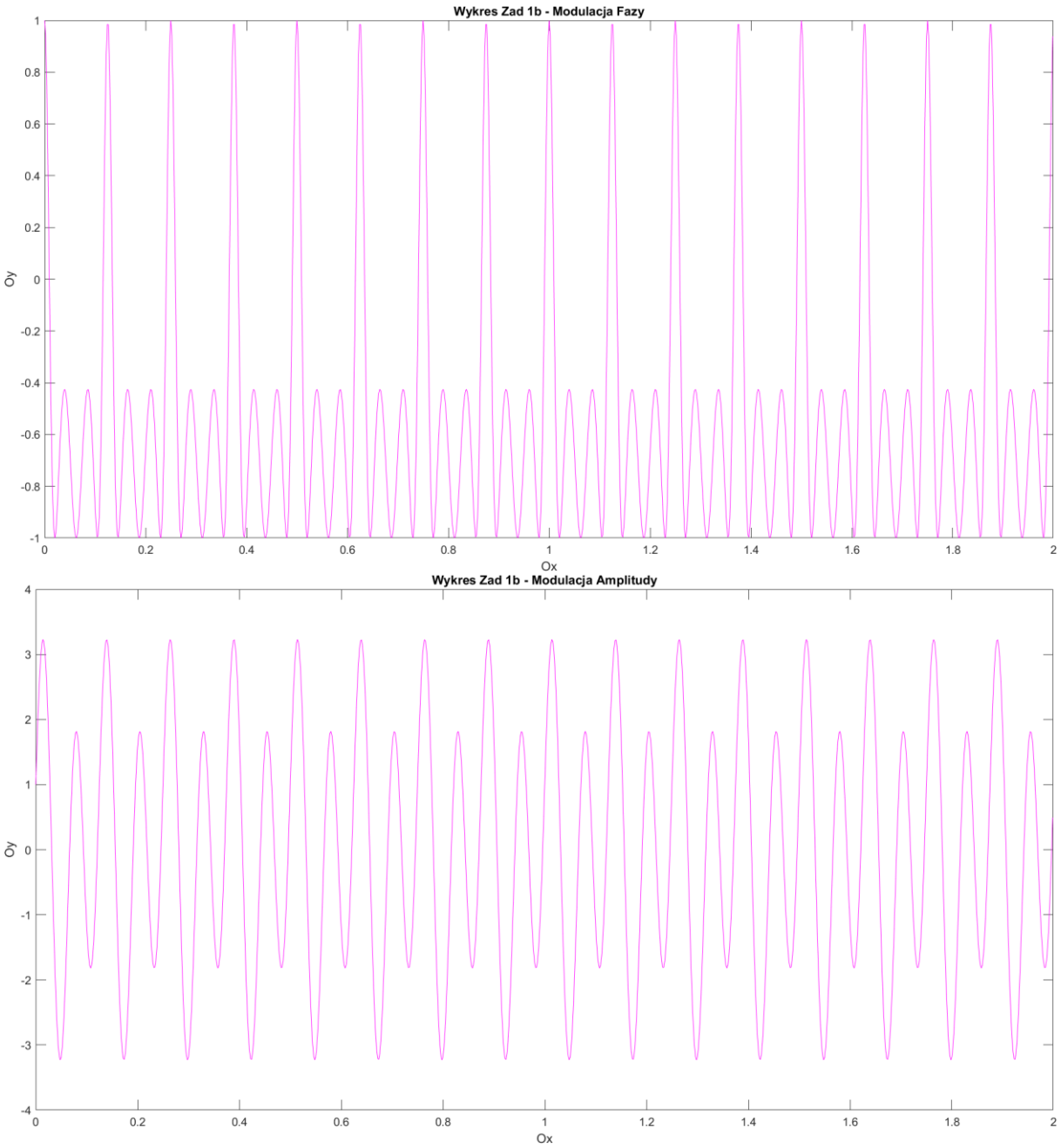

Wykres Zad 1 - Ton Prosty
Oy
Ox
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2


Wykres Zad 1 - Modulacja Amplitudy
Oy
Ox
1.5
1
0.5
0
-0.5
-1
-1.5
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2

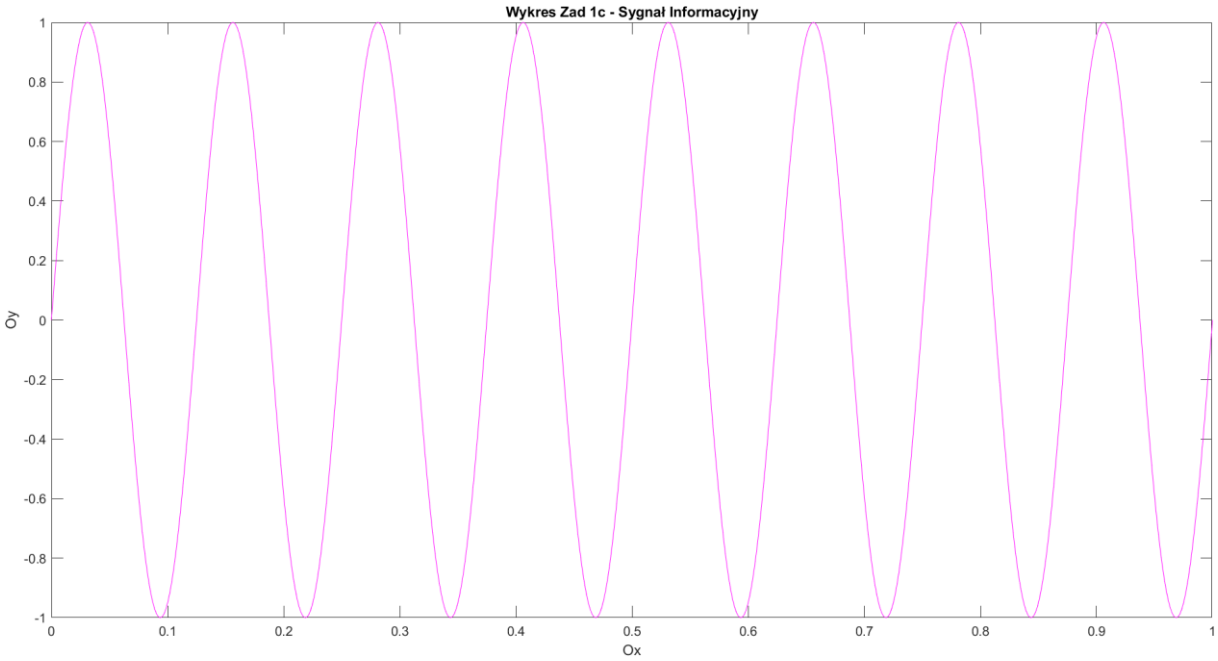

Wykres Zad 1 - Modulacja Fazy
Oy
Ox
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2

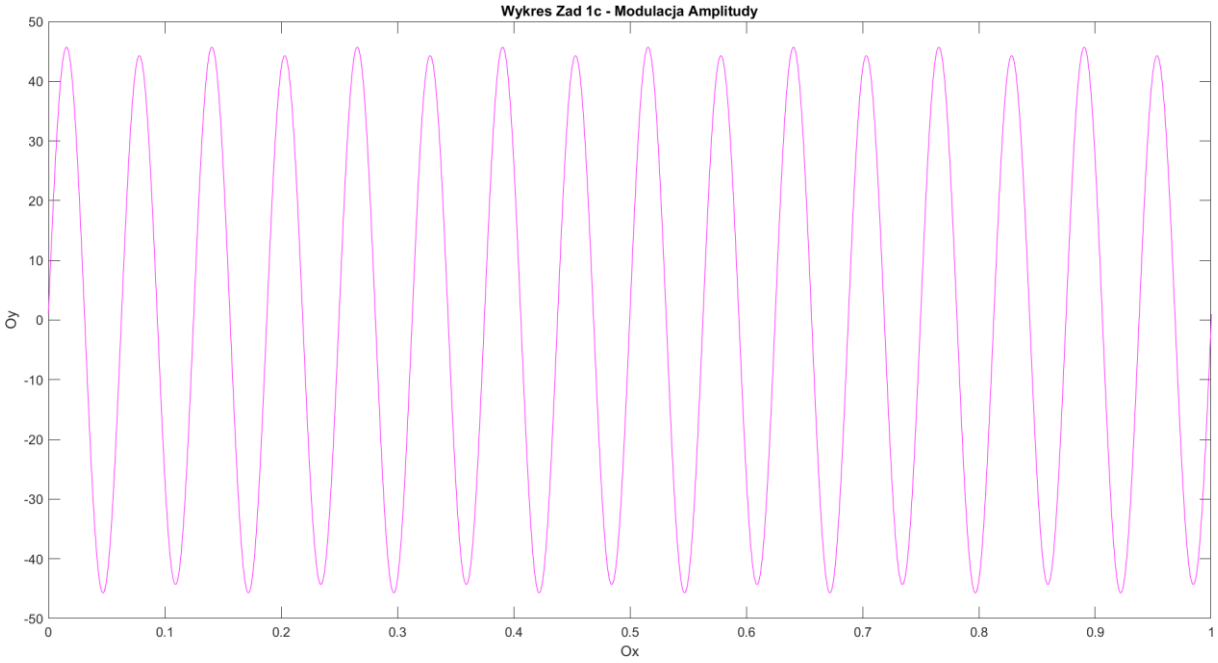

Wykres Zad 1 - Sygnał informacyjny
Oy
Ox
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2

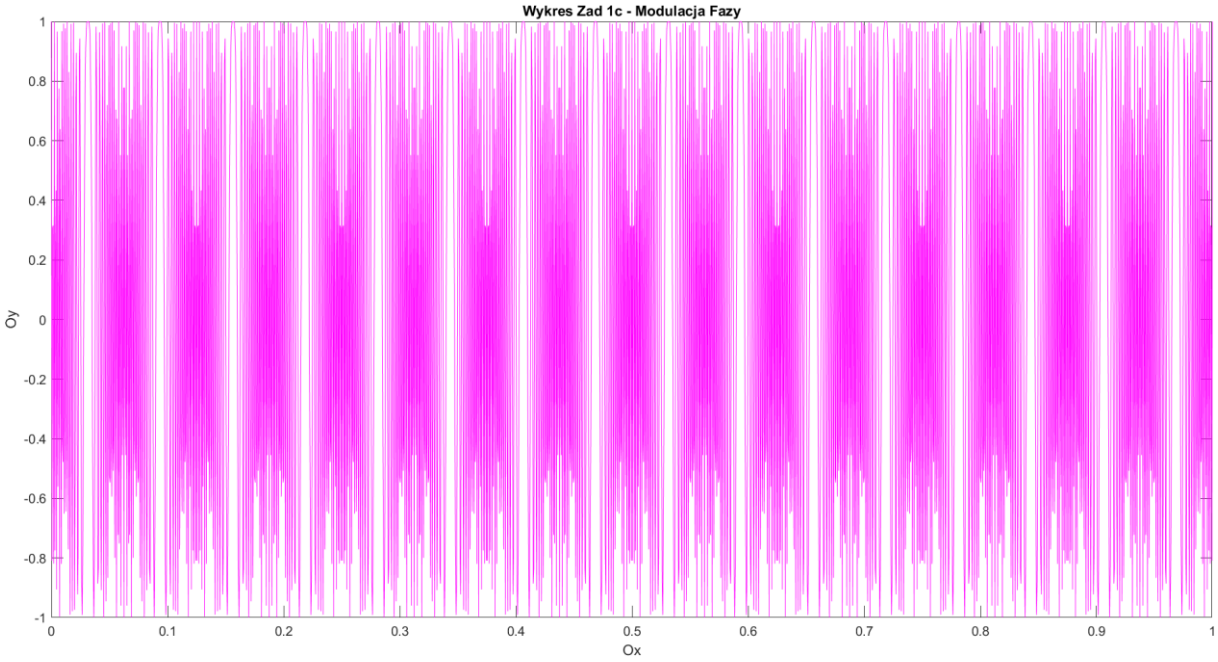
b)


Wykres Zad 1b - Modulacja Fazy
Oy
Ox
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2
Wykres Zad 1b - Modulacja Amplitudy
Oy
Ox
4
3
2
1
0
-1
-2
-3
-4
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2

c) Dla czytelności zmieniłem w tym podpunkcie zakres na wykresie. Załączam więc kolejny wykres sygnału informacyjnego.







<u>Zad 2 i 3:</u>

Kod do zadań 2 i 3:

```
#include <iostream>
#include <complex>
#include <fstream>

#define _USE_MATH_DEFINES

double pi = 3.14159265359;

using namespace std;

complex<double>* DFT(const double* tab, int N)
{
	complex<double>* tab2 = new complex<double>[N];

	for (int k = 0; k < N; k++)
	{
		tab2[k] = 0;
		complex<double> WN = cos(tab[k]) + 1i * sin(tab[k]);

		for (int n = 0; n < N; n++)
		{
			tab2[k] += tab[n] * pow(WN, -k * n);
		}
	}

	return tab2;
```

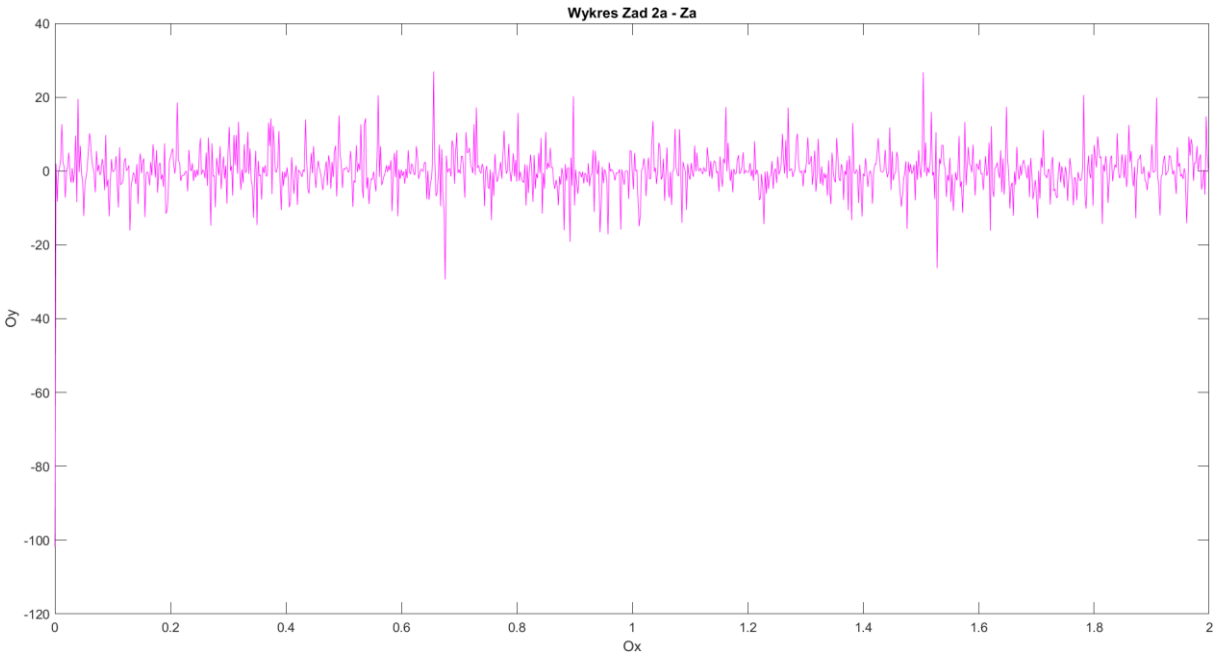
```
}

double ton_prosty(double a, double F, double phi, double t)
{
	double s = a * sin(2 * pi * F * t + phi);
	return s;
}

int main()
{
	double a = 1;//volty
	double A = 2;//z numeru albumu
	double F = 8;
	double phi = 2 * pi;
	double fs = 500;// (?)
	double Ts = 1 / fs;

	double kA = 0.5, kp = 1; //(a)
	//double kA = 10, kp = 3; //(b)
	//double kA = 90, kp = 99; //(c)

	ofstream saveOX("zad1OX.txt");
	ofstream saveTonProsty("zad1sig.txt");
	ofstream saveM("zad1M.txt");
	ofstream saveZa("zad1Za.txt");
	ofstream saveZp("zad1Zp.txt");

	int count = 0;

	for (double i = 0; i < A; i = i + Ts)
	{
		count++;
	}

	double* sig = new double[count];
	double* Za = new double[count];
	double* Zp = new double[count];
	int ilosc = count;
	count = 0;

	double fn = F;

	//edit zad 3

	double famin, famax, Wa;


	for (double i = 0; i < A; i = i + Ts)
```

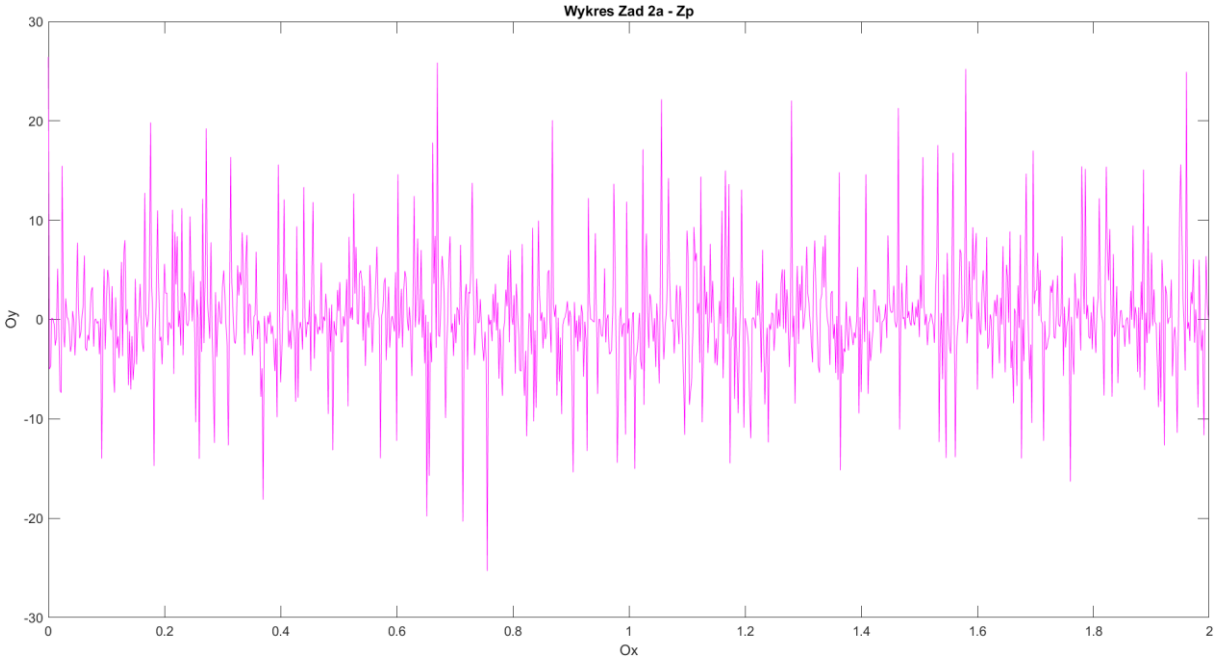
```
{
	sig[count] = ton_prosty(a, F, phi, i);
	saveOX << i << endl;
	saveTonProsty << sig[count] << endl;

	double m = a * sin(2 * pi * F * i);
	saveM << m << endl;
	Za[count] = (kA * m + 1) * cos(2 * pi * fn * i);
	if (Za[count] < -3)
		Za[count] = -3;
	saveZa << Za[count] << endl;

	if (count == 0)
	{
		famin = Za[count];
		famax = Za[count];
	}
	else
	{
		if (Za[count] < famin)
		{
			famin = Za[count];
		}
		if (Za[count] > famax)
		{
			famax = Za[count];
		}
	}

	Zp[count] = cos(2 * pi * fn * i + kp * m);
	if (Zp[count] < -3)
		Zp[count] = -3;
	saveZp << Zp[count] << endl;

	count++;
}

Wa = famax - famin;
cout << "Szerokosc pasma sygnalu: " << Wa << endl;

//a: Zmodulowana amplituda: 2.20183
//b: Zmodulowana amplituda: 8.7191
//c: Zmodulowana amplituda: 48.708

//zad2
complex<double>* DFTvalues = DFT(Zp, count);

ofstream saveSpectrum("zad2Spectrum.txt");
```

```
	ofstream saveMprim("zad2Mprim.txt");

	double* M = new double[ilosc];
	double* Mprim = new double[ilosc];

	for (int i = 0; i < count; i++)
	{
		M[i] = sqrt(pow(real(DFTvalues[i]), 2) + pow(imag(DFTvalues[i]), 2));
		saveSpectrum << M[i] << endl;
		Mprim[i] = 10 * log10(M[i]);
		saveMprim << Mprim[i] << endl;
	}

	//zamkniecie strumieni

	saveSpectrum.close();
	saveMprim.close();
	saveOX.close();
	saveTonProsty.close();
	saveM.close();
	saveZa.close();
	saveZp.close();

	return 0;
}
```

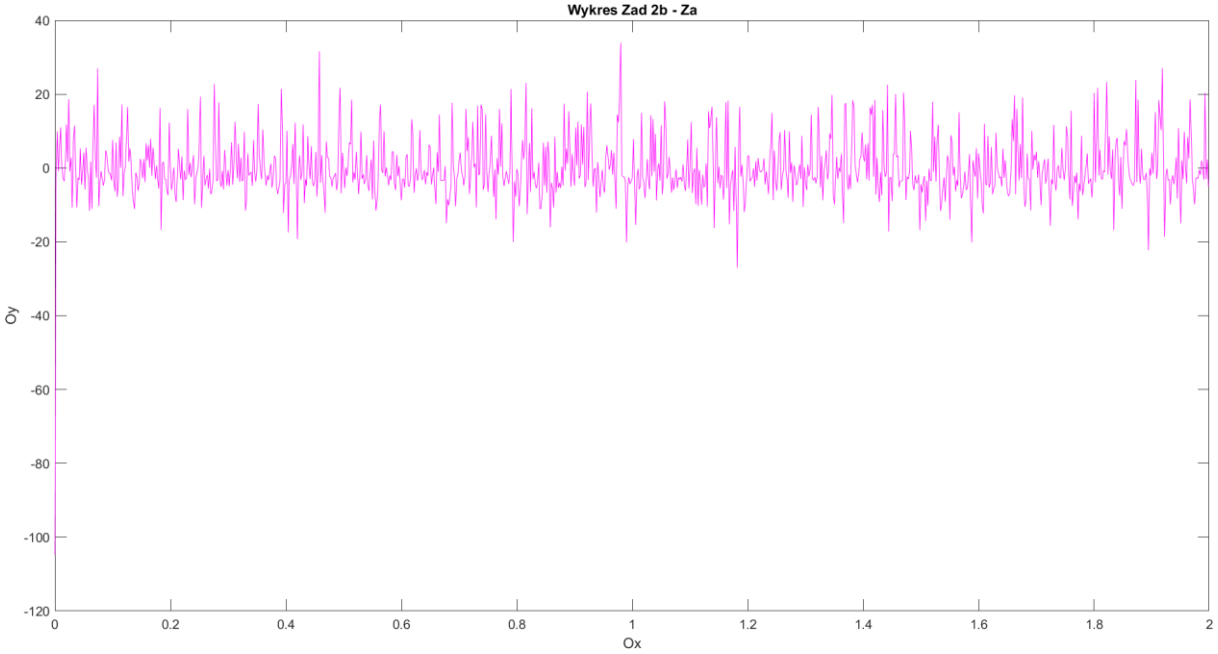
Wykresy:

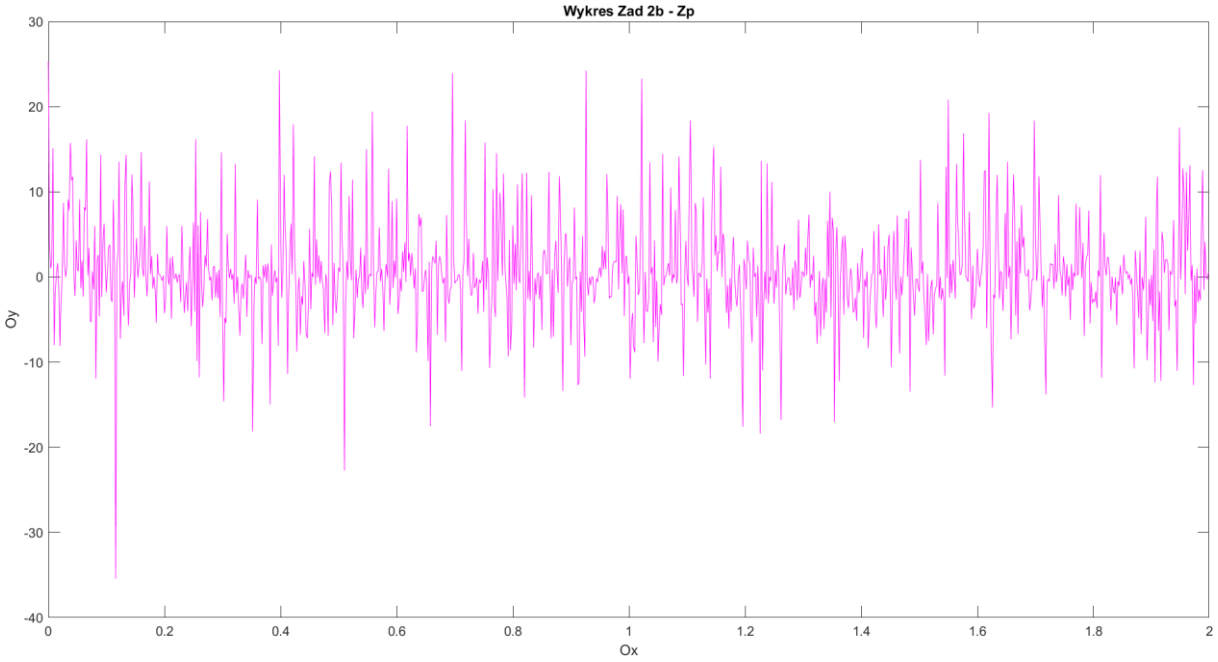
a)



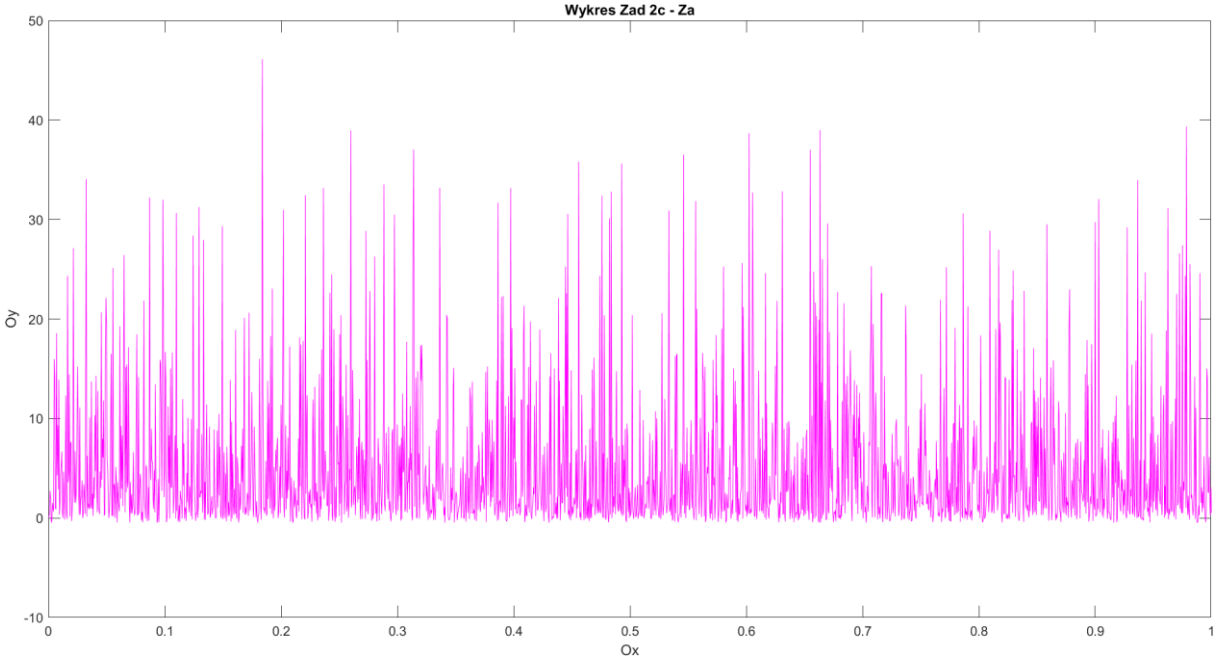

Wykres Zad 2a - Zp
Oy
Ox
30
20
10
0
-10
-20
-30
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2

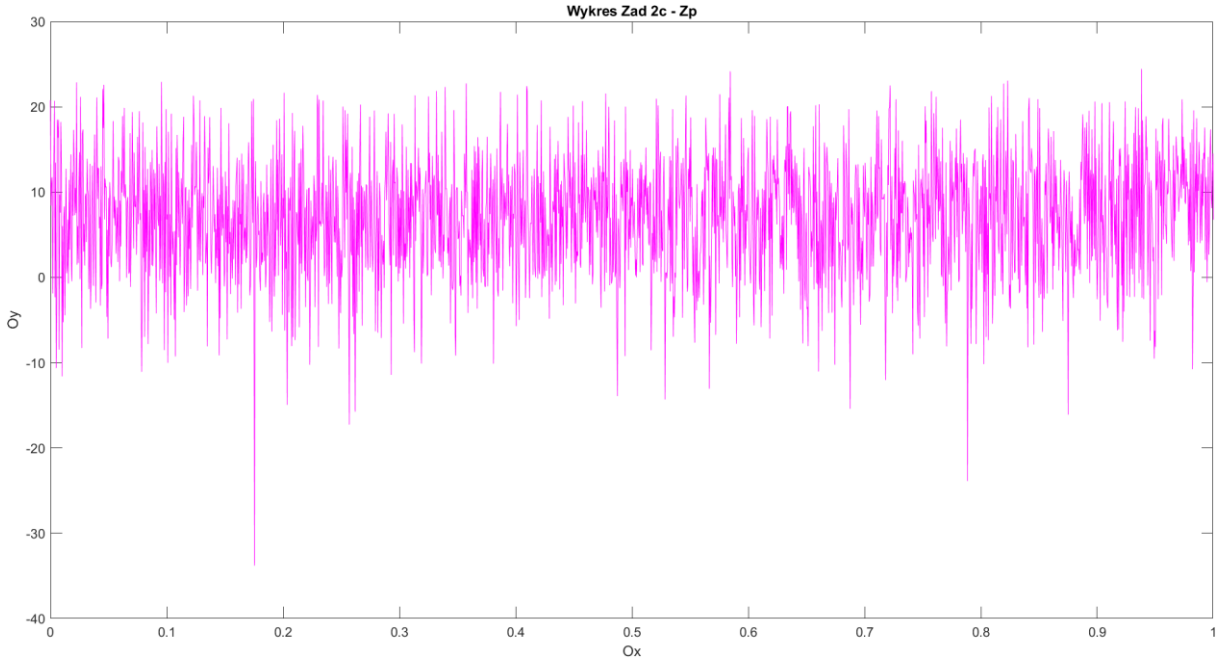
b)


Wykres Zad 2b - Za
Oy
Ox
40
20
0
-20
-40
-60
-80
-100
-120
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2


Wykres Zad 2b - Zp
30
20
10
0
-10
-20
-30
-40
Oy
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2
Ox

c)


Wykres Zad 2c - Za
50
40
30
20
10
0
-10
Oy
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
Ox


Wykres Zad 2c - Zp
30
20
10
0
-10
-20
-30
-40
Oy
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
Ox