1. Explain the property of linear independence and linear combination.

> Definition of linear independence

The polynomials ,$\{\Phi_0(t), \Phi_1(t), \ldots, \Phi_n(t)\}$, are said to be *linearly independent,* if the resultant of their *linear combination* is zero can only be satisfied by choosing all of coefficients zer

$$\text{s.t. } a_0\Phi_0(t) + a_1\Phi_1(t) + a_2\Phi_2(t) + \cdots + a_n\Phi_n(t) = 0 \Rightarrow a_0 = a_1 = \cdots = a_n = 0$$

즉, 선형 독립이 아니라면, 0이 아닌 계수가 존재하고, 하나의 다항식을 다른 다항식의 중첩(superposition[1])으로 표현이 가능하다는 것을 뜻한다. 따라서 이러한 1차 결합(선형 독립하지 않은 1차 결합)은 기저가 될 수 없다.

$$\Phi_0(t) = \frac{-a_1}{a_0}\Phi_0(t) + \frac{-a_2}{a_0}\Phi_2(t) + \cdots + \frac{-a_n}{a_0}\Phi_n(t)$$

> Definition of linear combination

The *linear combination* of those polynomials with those complex numbers as coefficients is

$$a_0\Phi_0(t) + a_1\Phi_1(t) + a_2\Phi_2(t) + \cdots + a_n\Phi_n(t)$$

1차 결합은 각 항의 덧셈과 상수의 곱셈으로만 표현된 식을 의미한다.

2. Write a C(or C++) code for Horner’s method for degree n polynomial. Execute the program and show an example using a real numbers.

```cpp
#include <iostream>

template<typename iter_Container>
 double Horner( iter_Container begin_Coefficient /* a_0 (the address of first coefficient) */,
                iter_Container end_Coefficient   /* a_n (the address of the container) */,
                double t                         /* independent variable */)
{
  double result_Polynominal = 0;
  while (end_Coefficient != begin_Coefficient) {
    result_Polynominal = result_Polynominal * t + *(--end_Coefficient);
  }
  return result_Polynominal;
}

int main()
{
  int degree_Polynominal = 0;
  std::cout << "Set the degree of polynominal(e.g. n): ";
  std::cin >>  degree_Polynominal;

  float* pCoefficients = new float[degree_Polynominal];
  std::cout << "Set coefficients in order of increasing dergree(e.g. a_0, a_1, ... , a_n)\n: ";
  for(auto ii = 0; ii < degree_Polynominal; ii++) {
    std::cin >> pCoefficients[ii];
  }
  std::cout << "Your coefficients: ";
  for(auto ii = 0; ii < degree_Polynominal; ii++) {
    std::cout << pCoefficients[ii] << ", ";
  }
  std::cout << "\n";

  float variable = 0;
  std::cout << "Set variable(e.g. t): ";
  std::cin >> variable;
  std::cout << "The result of Horner's Method is "
            << Horner(pCoefficients, pCoefficients + 4, variable)
            << std::endl;
  delete[] pCoefficients;
}
```

```
Terminal
Set the degree of polynominal(e.g. n): 4
Set coefficients in order of increasing dergree(e.g. a_0, a_1, ... , a_n)
: 3 2 1 2
Your coefficients: 3, 2, 1, 2,
Set variable(e.g. t): 2.5
The result of Horner's Method is 45.5
```

result

3. Do the following 2 equations have the same roots?

$$a(t) = 2 - 3t + t^2$$
$$b(t) = t^3 - t^2$$

`Resultant = 0` 식을 이용한다.

$$a(\tau) = b(\tau) = 0$$

$$a(\tau) = (\tau - 1)(\tau - 2) = 0$$
$$\therefore \tau = 1,\ \tau = 2$$

$$b(\tau) = \tau^2(\tau - 1) = 0$$
$$\therefore \tau = 0,\ \tau = 1$$

$a(1) = b(1) = 0$ 이므로, 공통근($t = 1$)을 가진다. □

---

1. $$F(\alpha x_1 + \beta x_2) = \alpha F(x_1) + \beta F(x_2) \quad \textbf{Superposition}$$

$$\begin{bmatrix} F(x_1 + x_2) = F(x_1) + F(x_2) & \textbf{Additivity} \\ F(\alpha x) = \alpha F(x) & \textbf{Homogeneity} \end{bmatrix}$$

↩