myNP.h

```
#include "../../include/myNP.h"
```

자신의 폴더에 위치에 있는 myNP.h라는 헤더파일 가져오는 방법

```
#define     PI      3.14159265358979323846264338327950288419716939937510582
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
```

PI 값 정의 실시

기본 library를 include 시키기

Integration

IntegrateRect()

적분을 구할 때 각 구간별로 직사각형의 넓이를 구한 후 구한 넓이를 합한다

1. datapoint의 개수는 m개이다.
2. 구간의 개수는 m-1개이다.
3. 각 구간별로 직사각형 넓이를 구한 후 구한 넓이를 다 합한다.

```
double IntegrateRect(double x[], double y[], int m);
```

Parameters

> x[]: x의 값들을 모아 놓은 배열
>
> y[]: y의 값들을 모아 놓은 배열
>
> m: x의 배열안에 들어있는 요소들의 개수를 의미한다. 이때 x와 y의 요소의 개수는 같아야 한다.

Example code

```
#include "../../include/myNP.h"

int main(){
    double x[] = { 0, 5, 10, 15, 20, 25, 30, 35, 40, 45, 50, 55, 60 };
    double y[] = { 0, 3, 8, 20, 33, 42, 40, 48, 60, 12, 8, 4, 3 };
    int M = sizeof(x) / sizeof(x[0]);
    double I_rect = IntegrateRect(x, y, M);
    printf("I_rect  = %f\n", I_rect);
    return 0;
}
```

output

```
I_rect=1390.0000000
```

Warning

- x, y는 double, m은 int의 형식이어야 한다.
- m은 x의 배열안에 들어있는 요소들의 개수를 의미한다. 이때 x와 y의 요소의 개수는 같아야 한다.

trapz()

적분을 구할 때 각 구간별로 사다리꼴의 넓이를 구한 후 구한 넓이를 합한다

1. datapoint의 개수는 m개이다.
2. 구간의 개수는 m-1개이다.
3. 각 구간별로 사다리꼴의 넓이를 구한 후 구한 넓이를 다 합한다.

```
double trapz(double x[], double y[], int m);
```

Parameters

> x[]: x의 값들을 모아 놓은 배열
>
> y[]: y의 값들을 모아 놓은 배열
>
> m: x의 배열안에 들어있는 요소들의 개수를 의미한다. 이때 x와 y의 요소의 개수는 같아야 한다.

Example code

```
#include "../../include/myNP.h"

int main(){
    double x[] = { 0, 5, 10, 15, 20, 25, 30, 35, 40, 45, 50, 55, 60 };
    double y[] = { 0, 3, 8, 20, 33, 42, 40, 48, 60, 12, 8, 4, 3 };
    int M = sizeof(x) / sizeof(x[0]);
    double I_trapz = trapz(x, y, M);
    printf("I_trapz = %f\n\n", I_trapz);
    return 0;
}
```

output

```
I_trapz = 1397.5000000
```

Warning

- x, y는 double, m은 int의 형식이어야 한다.
- m은 x의 배열안에 들어있는 요소들의 개수를 의미한다. 이때 x와 y의 요소의 개수는 같아야 한다.

simson13()

적분을 구할 때 각 구간별로 simson13의 방식을 이용해 넓이를 구한 후 구한 넓이를 합한다

$$I = \frac{h}{3}[f(x_0) + 4\sum_{i=1,3,5}^{N-1} f(x_i) + 2\sum_{k=2,4,6}^{N-2} f(x_k) + f(x_N)]$$

1. datapoint의 개수는 N+1개이다.
2. 구간의 개수는 N/2개이다.

```
double simpson13(double x[], double y[], int m);
```

Parameters

> x[]: x의 값들을 모아 놓은 배열
>
> y[]: y의 값들을 모아 놓은 배열
>
> m: x의 배열안에 들어있는 요소들의 개수를 의미한다. 이때 x와 y의 요소의 개수는 같아야 한다.

Example code

```
#include "../../include/myNP.h"

int main(){
    double x[] = { 0, 5, 10, 15, 20, 25, 30, 35, 40, 45, 50, 55, 60 };
    double y[] = { 0, 3, 8, 20, 33, 42, 40, 48, 60, 12, 8, 4, 3 };
    int M = sizeof(x) / sizeof(x[0]);
    double I_simpson13 = simpson13(x, y, M);
    printf("I_simpson13  = %f\n\n", I_simpson13);
    return 0;
}
```

output

```
I_simson13 = 1361.666667
```

Warning

- x, y는 double, m은 int의 형식이어야 한다.
- m은 x의 배열안에 들어있는 요소들의 개수를 의미한다. 이때 x와 y의 요소의 개수는 같아야 한다.
- m-1은 짝수여야 한다.

integral()

적분을 구할 때 각 구간별로 simson13, simson38의 방식을 이용해 넓이를 구한 후 구한 넓이를 합한다

1. 원하는 만큼 구간을 나눈다.
2. 구간의 개수가 짝수인 경우 simson13의 방식을 이용하여 적분을 실시한다.
3. 구간의 개수가 홀수이면서 3인 경우에는 simson38의 방식을 이용해 적분을 실시한다.
4. 구간의 개수가 홀수이면서 3보다 큰 경우에는 처음 x(0)~x(3)까지는 simson38 나머지 구간은 simson13의 방식을 이용해 적분을 실시한다.

```
double integral(double func(const double x), double a, double b, int n);
```

Parameters

> func(const double x): 적분하기를 원하는 함수
>
> a: 초기 설정한 두 구간 중 제일 작은 값
>
> b: 초기 설정한 두 구간 중 제일 큰 값
>
> n: 몇 개로 구간을 나눌 것인지 정해주는 정수

Example code

```
#include "../../include/myNP.h"
double myFunc(const double x) {
    double i = 0;
    i = sqrt(1 - (x * x));

    return i;
}
int main(){
    double a = -1;//처음값
    double b = 1;//나중값
    double N = 13;//구간의 개수
    double integ = 0;//초기값 설정
    integ= integral(myFunc, a, b, N);
    printf("Integral  = %f", integ);
    return 0;
}
```

output

```
Integral = 1.555333
```

Warning

- a, b는 double, n은 int의 형식이어야 한다.
- 적분을 원하는 함수도 같이 입력해주어야 한다.

integral1()

1. 적분을 구할 때 각 구간별로 simson13의 방식을 이용해 넓이를 구한 후 구한 넓이를 합한다.
2. 구간의 개수가 홀수라면 구간의 개수를 1증가시키고 simson13의 방식을 이용해서 적분을 구한다.

```
double integral1(double func(const double x), double a, double b, int n);
```

Parameters

> func(const double x): 적분하기를 원하는 함수
>
> a: 초기 설정한 두 구간 중 제일 작은 값
>
> b: 초기 설정한 두 구간 중 제일 큰 값
>
> n: 몇 개로 구간을 나눌 것인지 정해주는 정수

Example code

```
#include "../../include/myNP.h"
double myFunc(const double x) {
    double i = 0;
    i = sqrt(1 - (x * x));

    return i;
}
int main(){
    double a = -1;//처음값
    double b = 1;//나중값
    double N = 13;//구간의 개수
    double integ = 0;//초기값 설정
    integ= integral1(myFunc, a, b, N);
    printf("Integral  = %f", integ);
    return 0;
}
```

output

```
Integral = 1.558323
```

Warning

- a, b는 double, n은 int의 형식이어야 한다.
- 적분을 원하는 함수도 같이 입력해주어야 한다.

simson38()

적분을 구할 때 각 구간별로 simson38의 방식을 이용해 넓이를 구한 후 구한 넓이를 합한다

$$I = \frac{3h}{8}[f(x_0) + 3\sum_{i=1,4,7}^{N-2}[f(x_i) + f(x_{i+1})] + 2\sum_{k=3,6,9}^{N-3} f(x_k) + f(x_N)]$$

1. datapoint의 개수는 N+1개이다.
2. 구간의 개수는 N/3개이다.

```
double simpson38(double x[], double y[], int m);
```

Parameters

> x[]: x의 값들을 모아 놓은 배열
>
> y[]: y의 값들을 모아 놓은 배열

> m: x의 배열안에 들어있는 요소들의 개수를 의미한다. 이때 x와 y의 요소의 개수는 같아야 한다.

Example code

```
#include "../../include/myNP.h"

int main(){
    double x[] = { -3, -2.25, -1.5, -0.75};
    double y[] = { 0, 2.1875, 3.75, 4.6875 };
    int M = sizeof(x) / sizeof(x[0]);//data 사이즈 계산

    double I_simpson38 = 0;//초기값 설정
    I_simpson38 = simpson38(x, y, M);
    printf("I_simpson38  = %f", I_simpson38);
    return 0;
}
```

output

```
I_simson38=6.328125
```

Warning

- x, y는 double, m은 int의 형식이어야 한다.
- m은 x의 배열안에 들어있는 요소들의 개수를 의미한다. 이때 x와 y의 요소의 개수는 같아야 한다.
- m-1 즉 구간은 3의 배수여야 한다.