Name: 21700791 ID: 홍세현

**Output Display**

**Program Output Display**

|  |
| --- |
|  |

**Mid-Point Method**

Main() source file

|  |
| --- |
| // Mid Point Method for Discrete Data Input  double I\_midpoint = IntegralMid(x, y, M);  printf("I\_midpoint = %f\n\n", I\_midpoint); |

NM function source code

|  |
| --- |
| **myNM.cpp**  // Mid Point Method for Discrete Data Input  double IntegralMid(double \_x[], double \_y[], int \_m) {  // # of data : m  // # of interval : m-1  double Out = 0;  int N = \_m - 1; // Number of Intervals  for (int i = 0; i < N; i++)  Out += ( (\_y[i] + \_y[i + 1]) / 2 ) \* (\_x[i + 1] - \_x[i]);    return Out;  }  **myNM.h**  // Mid Point Method for Discrete Data Input  extern double IntegralMid(double \_x[], double \_y[], int \_m); |

**Trapezoidal Method**

Main() source file

|  |
| --- |
| // Trapezoidal Method for Discrete Data Input  double I\_trapz = trapz(x, y, M);  printf("I\_trapz = %f\n\n", I\_trapz); |

NM function source code

|  |
| --- |
| **myNM.cpp**  // Trapezoidal Method for Discrete Data Input  double trapz(double \_x[], double \_y[], int \_m){  // # of data : m  // # of interval : m-1  double Out = 0;  int N = \_m - 1; // Number of Intervals  for (int i = 0; i < N ; i++)  Out += ( \_y[i] + \_y[i + 1] ) \* ( \_x[i+1] - \_x[i] ) ;  Out = 0.5 \* Out;  return Out;  }  **myNM.h**  // Trapezoidal Method for Discrete Data Input  extern double trapz(double \_x[], double \_y[], int \_m); |

**Simpson 1/3 Method**

Main() source file

|  |
| --- |
| double myFunc(const double x) {  double y = sqrt(1 - pow(x, 2));  return y;  }  double a = -1; // 적분구간 a  double b = 1; // 적분구간 b  double n = 12; // Number of Intervals = 12  // Simpson 1/3 Method for Function Input  double I\_simpson13 = integral(myFunc, a, b, n);  printf("I\_simpson1/3 = %f\n\n", I\_simpson13); |

NM function source code

|  |
| --- |
| **myNM.cpp**  // Simpson 1/3 Method for Function Input  double integral(double func(const double \_x), double \_a, double \_b, int \_n) {  double Out = 0;  double temp1 = 0;  double temp2 = 0;  double h = (\_b - \_a) / \_n;  for (int i = 1; i < \_n ; i += 2){  double xi = \_a + (h \* i); // Interval에 맞춘 x좌표  temp1 += (4 \* func(xi));  }  for (int j = 2; j < \_n-1; j += 2){  double xi = \_a + (h \* j); // Interval에 맞춘 x좌표  temp2 += (2 \* func(xi));  }    Out = (h / 3) \* ( func(\_a) + temp1 + temp2 + func(\_b) );  return Out;  }  **myNM.h**  // Simpson 1/3 Method for Function Input  extern double integral(double func(const double \_x), double \_a, double \_b, int \_n); |

**Simpson 3/8 Method**

Main() source file

|  |
| --- |
| // Simpson 3/8 Method for Function Input  double I\_simpson38 = integral38(myFunc, a, b, n);  printf("I\_simpson3/8 = %f\n\n", I\_simpson38); |

NM function source code

|  |
| --- |
| **myNM.cpp**  // Simpson 3/8 Method for Function Input  double integral38(double func(const double \_x), double \_a, double \_b, int \_n) {  double Out = 0;  double temp1 = 0;  double temp2 = 0;  double h = (\_b - \_a) / \_n;  for (int i = 1; i < \_n-1; i += 3) {  double xi = \_a + (h \* i); // Interval에 맞춘 x좌표  temp1 += 3 \* ( func(xi) + func(xi+h) );  }  for (int j = 3; j < \_n - 2; j += 3) {  double xi = \_a + (h \* j); // Interval에 맞춘 x좌표  temp2 += 2 \* func(xi);  }  Out = ( (3 \* h) / 8 ) \* (func(\_a) + temp1 + temp2 + func(\_b));  return Out;  }  **myNM.h**  // Simpson 3/8 Method for Function Input  extern double integral38(double func(const double \_x), double \_a, double \_b, int \_n); |