-programs > 1program.cpp > ...

```cpp
/* write a program to computer mean and weighted mean of raw data */


#include <iostream>
#include <vector>
using namespace std;

int main() {
    int n;
    cout << "Enter the number of data points: ";
    cin >> n;

    vector<double> data(n), weights(n);
    cout << "Enter the data points: ";
    for (int i = 0; i < n; ++i) {
        cin >> data[i];
    }

    cout << "Enter the corresponding weights: ";
    for (int i = 0; i < n; ++i) {
        cin >> weights[i];
    }

    double sum_data = 0, sum_weights = 0, weighted_sum = 0;
    for (int i = 0; i < n; ++i) {
        sum_data += data[i];
        sum_weights += weights[i];
        weighted_sum += data[i] * weights[i];
    }

    double mean = sum_data / n;
    double weighted_mean = weighted_sum / sum_weights;

    cout << "Mean: " << mean << "\nWeighted Mean: " << weighted_mean << endl;
    return 0;
}
```

```
PS C:\Users\khush\Desktop> cd "c:\Users\khush\
\Math-programs\" ; if ($?) { g++ 1program.cpp
gram } ; if ($?) { .\1program }
Enter the number of data points: 4
Enter the data points: 3
2
1
6
Enter the corresponding weights: 5
4
2
1
Mean: 3
Weighted Mean: 2.58333
PS C:\Users\khush\Desktop\Math-programs>
```

h-programs > 2program.cpp > main()

```cpp
/* write a program to computer the mean and weighted mean of discrete series */

#include <iostream>
#include <vector>
using namespace std;

int main() {
    int n;
    cout << "Enter the number of data points: ";
    cin >> n;

    vector<double> x(n), f(n);
    cout << "Enter the x values: ";
    for (int i = 0; i < n; ++i) cin >> x[i];

    cout << "Enter the frequencies: ";
    for (int i = 0; i < n; ++i) cin >> f[i];

    double sum_x = 0, sum_fx = 0, sum_f = 0;
    for (int i = 0; i < n; ++i) {
        sum_x += x[i];
        sum_fx += x[i] * f[i];
        sum_f += f[i];
    }

    double mean = sum_x / n;
    double weighted_mean = sum_fx / sum_f;

    cout << "Mean: " << mean << "\nWeighted Mean: " << weighted_mean << endl;
    return 0;
}
```

```
PS C:\Users\khush\Desktop> cd "c:\Users\khush\Desk
\Math-programs\" ; if ($?) { g++ 2program.cpp -o 2
gram } ; if ($?) { .\2program }
Enter the number of data points: 4
Enter the x values: 6
5
3
9
Enter the frequencies: 5
1
3
9
Mean: 5.75
Weighted Mean: 6.94444
PS C:\Users\khush\Desktop\Math-programs>
```

n-programs > 3program.cpp > ...

```cpp
/* write a program to computer the mean and weighted mean of continous series */
#include <iostream>
#include <vector>
using namespace std;
int main() {
    int n;
    cout << "Enter the number of classes: ";
    cin >> n;
    vector<double> lower(n), upper(n), freq(n), weights(n), mid(n);
    for (int i = 0; i < n; ++i) {
        cout << "Class " << i + 1 << " (lower upper): ";
        cin >> lower[i] >> upper[i];
        mid[i] = (lower[i] + upper[i]) / 2;
    }
    cout << "Enter frequencies: ";
    for (int i = 0; i < n; ++i) cin >> freq[i];

    cout << "Enter weights: ";
    for (int i = 0; i < n; ++i) cin >> weights[i];

    double sum_mid_freq = 0, sum_freq = 0;
    double sum_mid_weight = 0, sum_weight = 0;

    for (int i = 0; i < n; ++i) {
        sum_mid_freq += mid[i] * freq[i];
        sum_freq += freq[i];
        sum_mid_weight += mid[i] * weights[i];
        sum_weight += weights[i];
    }
    cout << "Mean: " << sum_mid_freq / sum_freq
         << "\nWeighted Mean: " << sum_mid_weight / sum_weight << endl;
    return 0;
}
```

```
PS C:\Users\khush\Desktop\Math-programs> cd
> cd "c:\Users
\khush\Desktop\Math-programs\" ; if ($?) { g++ 3progr
am.cpp -o 3program } ; if ($?) { .\3program }
Enter the number of classes: 3
Class 1 (lower upper): 11
6
Class 2 (lower upper): 5
20
Class 3 (lower upper): 3
10
Enter frequencies: 5
7
9
Enter weights: 10
15
20
Mean: 8.97619
Weighted Mean: 8.94444
PS C:\Users\khush\Desktop\Math-programs>
```

ograms > 4program.cpp > main()

```cpp
/* Mode and Median of Raw Data */

#include <iostream>
#include <algorithm>
using namespace std;

int main() {
    int n;
    cout << "Enter number of elements: ";
    cin >> n;
    float arr[n];
    cout << "Enter elements: ";
    for(int i=0; i<n; i++) cin >> arr[i];
    sort(arr, arr + n);

    // Median
    float median = (n % 2 != 0) ? arr[n/2] : (arr[n/2 - 1]
    cout << "Median = " << median << endl;

    // Mode
    int max_count = 0, mode = arr[0], count = 1;
    for(int i=1; i<n; i++) {
        if(arr[i] == arr[i-1]) count++;
        else count = 1;
        if(count > max_count) {
            max_count = count;
            mode = arr[i];
        }
    }
    cout << "Mode = " << mode;
    return 0;
}
```

```
PS C:\Users\khush\Desktop\Math-programs> cd
> cd "c:\U
rams\" ; if ($?) { g++ 4program.cpp -o 4program }
Enter number of elements: 5
Enter elements: 1
3
5
7
9
Median = 5
Mode = 3
PS C:\Users\khush\Desktop\Math-programs>
```

-programs > 5program.cpp > main()

```cpp
/* Program to compute the median of discrete series (x, f) */

#include <iostream>
#include <vector>
#include <algorithm>
using namespace std;

int main() {
    vector<pair<int, int>> data = {{1, 2}, {2, 3}, {3, 4},
    {4, 5}};
    vector<int> values;
    for (auto& p : data) {
        for (int i = 0; i < p.second; ++i) {
            values.push_back(p.first);
        }
    }
    sort(values.begin(), values.end());
    int n = values.size();
    double median = (n % 2 == 0) ? (values[n/2 - 1] + values
    [n/2]) / 2.0 : values[n/2];
    cout << "Median: " << median << endl;
    return 0;
}
```

```
PS C:\Users\khush\Desktop> cd "c:\Users\khus
 ($?) { g++ 5program.cpp -o 5program } ; if
Median: 3
PS C:\Users\khush\Desktop\Math-programs>
```

rograms > 6program.cpp > main()

```cpp
/* Program to compute the median of continuous series */

#include <iostream>
#include <vector>
#include <algorithm>
using namespace std;

int main() {
    vector<pair<double, double>> intervals = {{10, 20}, {20, 30},
    {30, 40}};
    vector<int> freqs = {5, 10, 15};
    vector<double> values;
    for (size_t i = 0; i < intervals.size(); ++i) {
        double mid = (intervals[i].first + intervals[i].second) / 2;
        for (int j = 0; j < freqs[i]; ++j) {
            values.push_back(mid);
        }
    }
    sort(values.begin(), values.end());
    int n = values.size();
    double median = (n % 2 == 0) ? (values[n/2 - 1] + values[n/2]) /
    2.0 : values[n/2];
    cout << "Median: " << median << endl;
    return 0;
}
```

```
PS C:\Users\khush\Desktop\Math-programs> c
> cd "
p\Math-programs\" ; if ($?) { g++ 6program.cp
($?) { .\6program }
Median: 30
PS C:\Users\khush\Desktop\Math-programs>
```

n-programs > 7program.cpp > main()

```cpp
/*  Program to compute the mode of discrete series (x, f) */


#include <iostream>
#include <vector>
#include <map>
using namespace std;

int main() {
    vector<pair<int, int>> data = {{1, 2}, {2, 3}, {3, 4}, {4, 5}};
    map<int, int> freqMap;
    for (auto& p : data) {
        freqMap[p.first] = p.second;
    }
    int mode = 0, maxFreq = 0;
    for (auto& p : freqMap) {
        if (p.second > maxFreq) {
            maxFreq = p.second;
            mode = p.first;
        }
    }
    cout << "Mode: " << mode << endl;
    return 0;
}
```

```
PS C:\Users\khush\Desktop\Math-programs> cd
> cd
p\Math-programs\" ; if ($?) { g++ 7program.
($?) { .\7program }
Mode: 4
PS C:\Users\khush\Desktop\Math-programs>
```

-programs > 8program.cpp > main()

```cpp
/*  Program to compute the mode of continuous series */

#include <iostream>
#include <vector>
#include <map>
using namespace std;

int main() {
    vector<pair<double, double>> intervals = {{10, 20}, {20, 30},
    {30, 40}};
    vector<int> freqs = {5, 10, 15};
    map<double, int> freqMap;
    for (size_t i = 0; i < intervals.size(); ++i) {
        double mid = (intervals[i].first + intervals[i].second) / 2;
        freqMap[mid] = freqs[i];
    }
    double mode = 0;
    int maxFreq = 0;
    for (auto& p : freqMap) {
        if (p.second > maxFreq) {
            maxFreq = p.second;
            mode = p.first;
        }
    }
    cout << "Mode: " << mode << endl;
    return 0;
}
```

```
PS C:\Users\khush\Desktop\Math-programs> c
> cd
p\Math-programs\" ; if ($?) { g++ 8program.c
($?) { .\8program }
Mode: 35
PS C:\Users\khush\Desktop\Math-programs>
```

programs > 9program.cpp > main()

```cpp
/*  Program to compute the standard deviation and variance of
discrete series */

#include <iostream>
#include <vector>
#include <cmath>
using namespace std;

int main() {
    vector<pair<int, int>> data = {{1, 2}, {2, 3}, {3, 4}, {4, 5}};
    double sum = 0, sumSq = 0, totalFreq = 0;
    for (auto& p : data) {
        sum += p.first * p.second;
        sumSq += p.first * p.first * p.second;
        totalFreq += p.second;
    }
    double mean = sum / totalFreq;
    double variance = (sumSq / totalFreq) - (mean * mean);
    double stdDev = sqrt(variance);
    cout << "Variance: " << variance << endl;
    cout << "Standard Deviation: " << stdDev << endl;
    return 0;
}
```

```
PS C:\Users\khush\Desktop\Math-programs> cd
> cd
p\Math-programs\" ; if ($?) { g++ 9program.c
($?) { .\9program }
Variance: 1.12245
Standard Deviation: 1.05946
PS C:\Users\khush\Desktop\Math-programs>
```

programs > 10program.cpp > main()

```cpp
/* Program to compute the standard deviation and variance of
continuous series */

#include <iostream>
#include <vector>
#include <cmath>
using namespace std;

int main() {
    vector<pair<double, double>> intervals = {{10, 20}, {20, 30},
    {30, 40}};
    vector<int> freqs = {5, 10, 15};
    double sum = 0, sumSq = 0, totalFreq = 0;
    for (size_t i = 0; i < intervals.size(); ++i) {
        double mid = (intervals[i].first + intervals[i].second) / 2;
        sum += mid * freqs[i];
        sumSq += mid * mid * freqs[i];
        totalFreq += freqs[i];
    }
    double mean = sum / totalFreq;
    double variance = (sumSq / totalFreq) - (mean * mean);
    double stdDev = sqrt(variance);
    cout << "Variance: " << variance << endl;
    cout << "Standard Deviation: " << stdDev << endl;
    return 0;
}
```

```
PS C:\Users\khush\Desktop> cd "c:\Users\khush
s\" ; if ($?) { g++ 10program.cpp -o 10progra
rogram }
Variance: 55.5556
Standard Deviation: 7.45356
PS C:\Users\khush\Desktop\Math-programs>
```

-programs > 11program.cpp > main()

```cpp
/* Program to compute the correlation using Karl Pearson's Correlation
*/

#include <iostream>
#include <vector>
#include <cmath>
using namespace std;

int main() {
    vector<double> x = {1, 2, 3, 4};
    vector<double> y = {2, 4, 6, 8};
    double sumX = 0, sumY = 0, sumXY = 0, sumX2 = 0, sumY2 = 0;
    int n = x.size();
    for (int i = 0; i < n; ++i) {
        sumX += x[i];
        sumY += y[i];
        sumXY += x[i] * y[i];
        sumX2 += x[i] * x[i];
        sumY2 += y[i] * y[i];
    }
    double numerator = sumXY - (sumX * sumY / n);
    double denominator = sqrt((sumX2 - (sumX * sumX / n)) * (sumY2 -
    (sumY * sumY / n)));
    double correlation = numerator / denominator;
    cout << "Correlation: " << correlation << endl;
    return 0;
}
```

```
PS C:\Users\khush\Desktop\Math-programs> c
> cd "c:\Users\khus
ktop\Math-programs\" ; if ($?) { g++ 11program.cpp -o 11pr
 } ; if ($?) { .\11program }
Correlation: 1
PS C:\Users\khush\Desktop\Math-programs>
```

-programs > 12program.cpp > main()

```cpp
/* Program to compute the regression coefficients */

#include <iostream>
#include <vector>
#include <cmath>
using namespace std;

int main() {
    vector<double> x = {1, 2, 3, 4};
    vector<double> y = {2, 4, 6, 8};
    double sumX = 0, sumY = 0, sumXY = 0, sumX2 = 0;
    int n = x.size();
    for (int i = 0; i < n; ++i) {
        sumX += x[i];
        sumY += y[i];
        sumXY += x[i] * y[i];
        sumX2 += x[i] * x[i];
    }
    double b1 = (n * sumXY - sumX * sumY) / (n * sumX2 - sumX * sumX);
    double b0 = (sumY - b1 * sumX) / n;
    cout << "Regression coefficients: b0 = " << b0 << ", b1 = " << b1
    << endl;
    return 0;
}
```

```
PS C:\Users\khush\Desktop\Math-programs> cd
> cd "c:\
ktop\Math-programs\" ; if ($?) { g++ 12program.c
 } ; if ($?) { .\12program }
Regression coefficients: b0 = 0, b1 = 2
PS C:\Users\khush\Desktop\Math-programs>
```

-programs > 13program.cpp > bisection(double, double, double)

```cpp
/* Program for Bisection method */

#include <iostream>
#include <cmath>
using namespace std;

double f(double x) {
    return x * x - 4;
}

double bisection(double a, double b, double tol) {
    if (f(a) * f(b) >= 0) {
        cout << "Invalid interval" << endl;
        return 0;
    }
    double c = a;
    while ((b - a) >= tol) {
        c = (a + b) / 2;
        if (f(c) == 0.0) break;
        else if (f(c) * f(a) < 0) b = c;
        else a = c;
    }
    return c;
}

int main() {
    double a = 0, b = 3, tol = 0.001;
    double root = bisection(a, b, tol);
    cout << "Root: " << root << endl;
    return 0;
}
```

```
PS C:\Users\khush\Desktop\Math-programs> c
> cd "c:\Users\khush\
ktop\Math-programs\" ; if ($?) { g++ 13program.cpp -o 13prog
 ] ; if ($?) { .\13program }
Root: 2.00024
PS C:\Users\khush\Desktop\Math-programs>
```

programs > 14program.cpp > main()

```cpp
/*  Program for Regula-falsi method */

#include <iostream>
#include <cmath>
using namespace std;

double f(double x) {
    return x * x - 4;
}

double regulaFalsi(double a, double b, double tol) {
    if (f(a) * f(b) >= 0) {
        cout << "Invalid interval" << endl;
        return 0;
    }
    double c = a;
    while (abs(f(c)) >= tol) {
        c = (a * f(b) - b * f(a)) / (f(b) - f(a));
        if (f(c) == 0.0) break;
        else if (f(c) * f(a) < 0) b = c;
        else a = c;
    }
    return c;
}

int main() {
    double a = 0, b = 3, tol = 0.001;
    double root = regulaFalsi(a, b, tol);
    cout << "Root: " << root << endl;
    return 0;
}
```

```
PS C:\Users\khush\Desktop\Math-programs> c
> cd "c:\Users\khush\Desktop\Math-programs\" ; if ($?) { g++ 14program.cpp -o 14program } ; if ($?) { .\14program }
Root: 1.99995
PS C:\Users\khush\Desktop\Math-programs>
```

```cpp
/* Program for Secant method */

#include <iostream>
#include <cmath>
using namespace std;

double f(double x) {
    return x * x - 4;
}

double secant(double x0, double x1, double tol) {
    double x2;
    while (abs(x1 - x0) >= tol) {
        x2 = x1 - (f(x1) * (x1 - x0)) / (f(x1) - f(x0));
        x0 = x1;
        x1 = x2;
    }
    return x1;
}

int main() {
    double x0 = 1, x1 = 3, tol = 0.001;
    double root = secant(x0, x1, tol);
    cout << "Root: " << root << endl;
    return 0;
}
```

```
rams\" ; if ($?) { g++ 15program.cpp -o 15program } ; if (
 .\15program }
Root: 2
PS C:\Users\khush\Desktop\Math-programs>
```

-programs > 16program.cpp > main()

```cpp
/* Program for Newton-Raphson method */

#include <iostream>
#include <cmath>
using namespace std;

double f(double x) {
    return x * x - 4;
}

double df(double x) {
    return 2 * x;
}

double newtonRaphson(double x, double tol) {
    double h = f(x) / df(x);
    while (abs(h) >= tol) {
        h = f(x) / df(x);
        x = x - h;
    }
    return x;
}

int main() {
    double x = 3, tol = 0.001;    ✨ Press Tab to accept your first sugg
    double root = newtonRaphson(x, tol);
    cout << "Root: " << root << endl;
    return 0;
}
```

```
PS C:\Users\khush\Desktop\Math-programs> c



                                        > cd "c:\Users
ktop\Math-programs\" ; if ($?) { g++ 16program.cpp -o
 } ; if ($?) { .\16program }
Root: 2
PS C:\Users\khush\Desktop\Math-programs>
```

rograms > 17program.cpp > gaussElimination(vector<vector<double>>&)

```cpp
/* Program for Gauss-Elimination method */

#include <iostream>
#include <vector>
using namespace std;

void gaussElimination(vector<vector<double>>& mat) {
    int n = mat.size();
    for (int i = 0; i < n; ++i) {
        for (int k = i + 1; k < n; ++k) {
            double factor = mat[k][i] / mat[i][i];
            for (int j = i; j <= n; ++j) {
                mat[k][j] -= factor * mat[i][j];
            }
        }
    }
    vector<double> x(n);
    for (int i = n - 1; i >= 0; --i) {
        x[i] = mat[i][n] / mat[i][i];
        for (int k = i - 1; k >= 0; --k) {
            mat[k][n] -= mat[k][i] * x[i];
        }
    }
    for (int i = 0; i < n; ++i) {
        cout << "x" << i + 1 << " = " << x[i] << endl;
    }
}

int main() {
    vector<vector<double>> mat = {{2, 1, -1, 8}, {-3, -1, 2, -11},
    {-2, 1, 2, -3}};
    gaussElimination(mat);
    return 0;
}
```

```
PS C:\Users\khush\Desktop\Math-programs> c
> cd "c:\Users\k
ktop\Math-programs\" ; if ($?) { g++ 17program.cpp -o 1
 } ; if ($?) { .\17program }
x1 = 2
x2 = 3
x3 = -1
PS C:\Users\khush\Desktop\Math-programs>
```

programs > 18program.cpp > lagrangeInterpolation(vector<double>&, vector<double>&, double)

```cpp
/* Program for Lagrange's Interpolation method */

#include <iostream>
#include <vector>
using namespace std;

double lagrangeInterpolation(vector<double>& x, vector<double>& y,
double xi) {
    double result = 0.0;
    for (size_t i = 0; i < x.size(); ++i) {
        double term = y[i];
        for (size_t j = 0; j < x.size(); ++j) {
            if (j != i) {
                term = term * (xi - x[j]) / (x[i] - x[j]);
            }
        }
        result += term;
    }
    return result;
}

int main() {
    vector<double> x = {1, 2, 3};
    vector<double> y = {1, 4, 9};
    double xi = 2.5;
    double yi = lagrangeInterpolation(x, y, xi);
    cout << "Interpolated value at " << xi << " is " << yi << endl;
    return 0;
}
```

```
PS C:\Users\khush\Desktop> cd "c:\Users\khush
ams\" ; if ($?) { g++ 18program.cpp -o 18prog
\18program }
Interpolated value at 2.5 is 6.25
PS C:\Users\khush\Desktop\Math-programs>
```

programs > 19program.cpp > newtonInterpolation(vector<double>&, vector<double>&, double)

```cpp
/* Program for Newton-Interpolation method */

#include <iostream> ...
using namespace std;

double newtonInterpolation(vector<double>& x, vector<double>& y, double xi) {
    int n = x.size();
    vector<vector<double>> diffTable(n, vector<double>(n));
    for (int i = 0; i < n; ++i) {
        diffTable[i][0] = y[i];
    }
    for (int j = 1; j < n; ++j) {
        for (int i = 0; i < n - j; ++i) {
            diffTable[i][j] = (diffTable[i + 1][j - 1] - diffTable[i][j - 1]) /
            (x[i + j] - x[i]);
        }
    }
    double result = diffTable[0][0];
    double term = 1.0;
    for (int i = 1; i < n; ++i) {
        term *= (xi - x[i - 1]);
        result += term * diffTable[0][i];
    }
    return result;
}

int main() {
    vector<double> x = {1, 2, 3};
    vector<double> y = {1, 4, 9};
    double xi = 2.5;
    double yi = newtonInterpolation(x, y, xi);
    cout << "Interpolated value at " << xi << " is " << yi << endl;
    return 0;
}
```

```
PS C:\Users\khush\Desktop\Math-programs>
> cd "c:
\khush\Desktop\Math-programs\" ; if ($?) { g++
ram.cpp -o 19program } ; if ($?) { .\19program
Interpolated value at 2.5 is 6.25
PS C:\Users\khush\Desktop\Math-programs>
```