



Projeto Computacional



Cecilia, Ingrid, Jamilly e Kamily

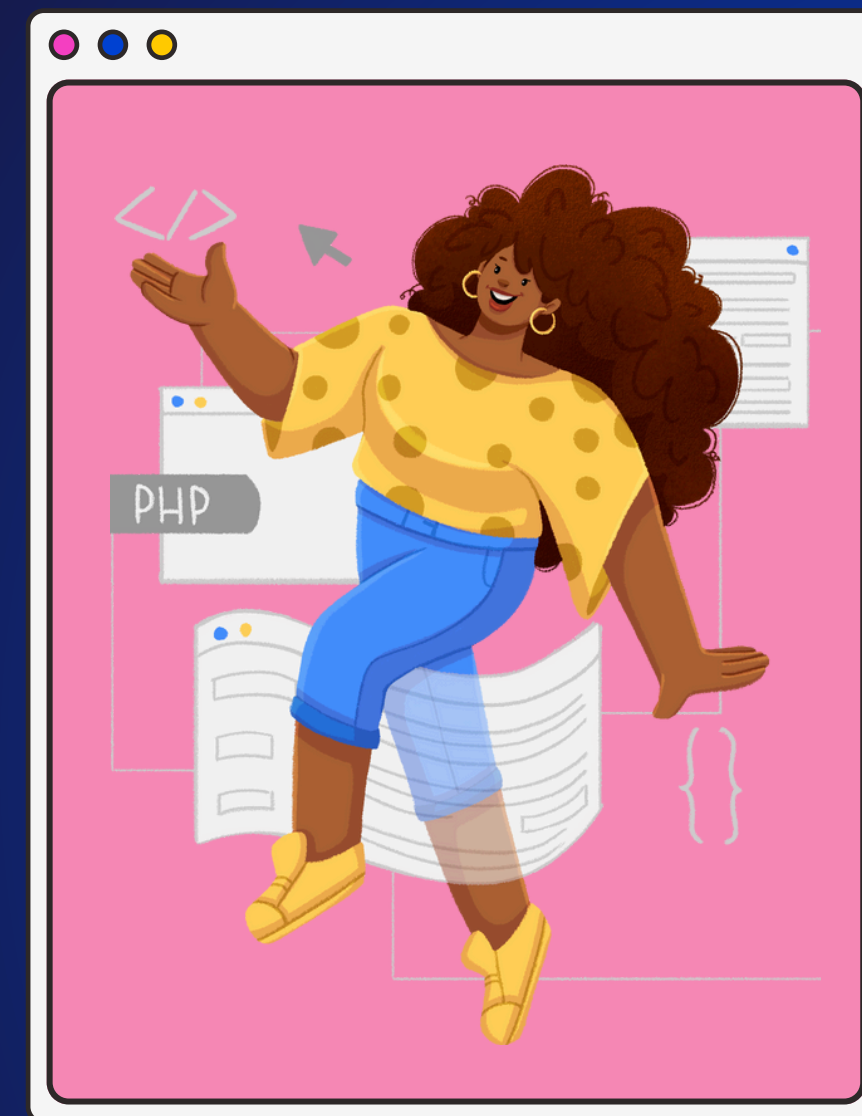


LOG IN >

apresentação

O nosso projeto foi criado com o objetivo de criar um programa que otimizasse o processo dos cálculos feitos em física experimental.

Visando isso, criamos uma calculadora. Ela contém as operações que envolvem propagação de erro - Desde as operações base de física experimental até cálculos mais trabalhosos que exigiriam um bom tempo resolvendo.



menu



Erro aleatório

1 

Soma de medidas

2 

Subtração de medidas

3 

Multiplicação de medidas

4 

Divisão de medidas

5 

Potencia de uma medida

6 

Multiplicação por escalar de
uma medida

7 


Discrepância de medidas

8 

funcionalidades



**realizar contas base no
contexto de física
experimental**



**realizar contas mais
trabalhosas, como calculo
de discrepância e erro
aleatório**



**Realiza operações
repetidamente
para aplicação em
massa**



**útil para aferir os
cálculos feitos
manualmente**

conteúdos utilizados

Funções para organizar em blocos as formulas necessárias para cada calculo

Condicional para realizar a operação dependendo da opção desejada

Loop para armazenar os resultados, para imprimir posteriormente e para ajudar na funcionalidade do menu

Struct (registros) utilizadas para agrupar diferentes tipos de dados relacionados nas operações.

desenvolvimento



**criação de menu
simples para ajudar
o usuário**



**desenvolvimento,
conversão e
programação das
formulas para cada um
das operações**



**implementação de
vetores, loops e structs
para o funcionamento
do programa**



**definição da função
principal e criação
das mensagens
impressas**



**processo de
testes e correção
de erros**

Bibliotecas utilizadas



iostream:
manipulação de
fluxo de dados
padrão do sistema



Vector:
modelo de classe
para contêineres
de sequência.



Cmath:
suporte a um
grande número de
funções
matemáticas úteis



Menu



```
void menu() {  
    cout << "Escolha uma opcao:" << endl;  
    cout << "1 - Erro aleatorio" << endl;  
    cout << "2 - Soma de medidas" << endl;  
    cout << "3 - Subtracao de medidas" << endl;  
    cout << "4 - Multiplicacao de medidas" << endl;  
    cout << "5 - Divisao de medidas" << endl;  
    cout << "6 - Potencia de uma medida" << endl;  
    cout << "7 - Multiplicacao por escalar de uma medida" << endl;  
    cout << "8 - Discrepancia de medidas" << endl;  
    cout << "" << endl;  
    cout << "Digite a opcao desejada:" << endl;  
}
```


Erro aleatório



$$\sigma = \sqrt{\sum_{i=1}^N \frac{(x_i - \bar{x})^2}{(N-1)}}$$

```
void ErroAleatorio() {
    int n;
    cout << "Digite o numero de medidas:" << endl;
    cin >> n;

    vector<Medida> medidas(n);
    double soma = 0, media, sdv = 0, dp, dpm;

    for (int i = 0; i < n; i++) {
        cout << "Digite a medida " << i + 1 << ":" << endl;
        cin >> medidas[i].valor;
        soma += medidas[i].valor;
    }

    media = soma / n;
    cout << "Esta eh a media das medidas: " << media << endl;

    for (int i = 0; i < n; i++) {
        sdv += pow(medidas[i].valor - media, 2);
    }
    dp = sqrt(sdv / (n - 1));
    cout << "Este eh o desvio padrao: " << dp << endl;

    dpm = dp / sqrt(n);
    cout << "Este eh o desvio padrao da media/erro aleatorio: " << dpm << endl;
}
```



Soma de medidas



```
void Soma() {  
    int vezes;  
    cout << "Digite quantas vezes deseja realizar a operacao:" << endl;  
    cin >> vezes;  
  
    for (int k = 0; k < vezes; k++) {  
        Medida m1, m2;  
        cout << "Digite a melhor estimativa da primeira medida:" << endl;  
        cin >> m1.valor;  
        cout << "Digite a incerteza da primeira medida:" << endl;  
        cin >> m1.incerteza;  
        cout << "Digite a melhor estimativa da segunda medida:" << endl;  
        cin >> m2.valor;  
        cout << "Digite a incerteza da segunda medida:" << endl;  
        cin >> m2.incerteza;  
  
        double MelhorEstimativa = m1.valor + m2.valor;  
        double IncertezaFinal = m1.incerteza + m2.incerteza;  
  
        cout << "Resultado da operacao " << k + 1 << ":" << endl;  
        cout << MelhorEstimativa << " +/- " << IncertezaFinal << endl;  
    }  
}
```

Subtração de medidas



```
void Subtracao() {
    int vezes;
    cout << "Digite quantas vezes deseja realizar a operacao:" << endl;
    cin >> vezes;

    for (int k = 0; k < vezes; k++) {
        Medida m1, m2;
        cout << "Digite a melhor estimativa da primeira medida:" << endl;
        cin >> m1.valor;
        cout << "Digite a incerteza da primeira medida:" << endl;
        cin >> m1.incerteza;
        cout << "Digite a melhor estimativa da segunda medida:" << endl;
        cin >> m2.valor;
        cout << "Digite a incerteza da segunda medida:" << endl;
        cin >> m2.incerteza;

        double MelhorEstimativa = m1.valor - m2.valor;
        double IncertezaFinal = m1.incerteza + m2.incerteza;

        cout << "Resultado da operacao " << k + 1 << ":" << endl;
        cout << MelhorEstimativa << " +/- " << IncertezaFinal << endl;
    }
}
```



```
void Multiplicacao() {
    int vezes;
    cout << "Digite quantas vezes deseja realizar a operacao:" << endl;
    cin >> vezes;

    for (int k = 0; k < vezes; k++) {
        Medida m1, m2;
        cout << "Digite a melhor estimativa da primeira medida:" << endl;
        cin >> m1.valor;
        cout << "Digite a incerteza da primeira medida:" << endl;
        cin >> m1.incerteza;
        cout << "Digite a melhor estimativa da segunda medida:" << endl;
        cin >> m2.valor;
        cout << "Digite a incerteza da segunda medida:" << endl;
        cin >> m2.incerteza;

        double MelhorEstimativa = m1.valor * m2.valor;
        double IncertezaFinal = (m1.valor * m2.incerteza) + (m2.valor * m1.incerteza);

        cout << "Resultado da operacao " << k + 1 << ":" << endl;
        cout << MelhorEstimativa << " +/- " << IncertezaFinal << endl;
    }
}
```



```
void Divisao() {
    int vezes;
    cout << "Digite quantas vezes deseja realizar a operacao:" << endl;
    cin >> vezes;

    for (int k = 0; k < vezes; k++) {
        Medida m1, m2;
        cout << "Digite a melhor estimativa da primeira medida:" << endl;
        cin >> m1.valor;
        cout << "Digite a incerteza da primeira medida:" << endl;
        cin >> m1.incerteza;
        cout << "Digite a melhor estimativa da segunda medida:" << endl;
        cin >> m2.valor;
        cout << "Digite a incerteza da segunda medida:" << endl;
        cin >> m2.incerteza;

        double MelhorEstimativa = m1.valor / m2.valor;
        double IncertezaFinal = (m1.incerteza / m2.valor) + ((m1.valor / (m2.valor * m2.valor)) * m2.incerteza);

        cout << "Resultado da operacao " << k + 1 << ":" << endl;
        cout << MelhorEstimativa << " +/- " << IncertezaFinal << endl;
    }
}
```



```
void potencia() {
    int vezes;
    cout << "Digite quantas vezes deseja realizar a operacao:" << endl;
    cin >> vezes;

    for (int k = 0; k < vezes; k++) {
        Medida m1;
        double expoente;
        cout << "Digite a melhor estimativa da medida:" << endl;
        cin >> m1.valor;
        cout << "Digite a incerteza da medida:" << endl;
        cin >> m1.incerteza;
        cout << "Digite o expoente:" << endl;
        cin >> expoente;

        double MelhorEstimativa = pow(m1.valor, expoente);
        double IncertezaFinal = expoente * pow(m1.valor, expoente - 1) * m1.incerteza;

        cout << "Resultado da operacao " << k + 1 << ":" << endl;
        cout << MelhorEstimativa << " +/- " << IncertezaFinal << endl;
    }
}
```


Multiplicação por escalar de uma medida

7



$$(aB) \pm (aDB) = a(B \pm DB)$$

```
void MultPorEscalar() {
    int vezes;
    cout << "Digite quantas vezes deseja realizar a operacao:" << endl;
    cin >> vezes;

    for (int k = 0; k < vezes; k++) {
        Medida m1;
        double escalar;
        cout << "Digite a melhor estimativa da medida:" << endl;
        cin >> m1.valor;
        cout << "Digite a incerteza da medida:" << endl;
        cin >> m1.incerteza;
        cout << "Digite o escalar:" << endl;
        cin >> escalar;

        double MelhorEstimativa = m1.valor * escalar;
        double IncertezaFinal = m1.incerteza * escalar;

        cout << "Resultado da operacao " << k + 1 << ":" << endl;
        cout << MelhorEstimativa << " +/- " << IncertezaFinal << endl;
    }
}
```



Definição

É a diferença entre duas medidas e só irá ser significativa se o intervalo de erros se sobreporem.

```
void discrepancia() {
    int vezes;
    cout << "Digite quantas vezes deseja realizar a operacao:" << endl;
    cin >> vezes;

    for (int k = 0; k < vezes; k++) {
        Medida m1, m2;
        cout << "Digite a melhor estimativa da primeira medida:" << endl;
        cin >> m1.valor;
        cout << "Digite a incerteza da primeira medida:" << endl;
        cin >> m1.incerteza;
        cout << "Digite a melhor estimativa da segunda medida:" << endl;
        cin >> m2.valor;
        cout << "Digite a incerteza da segunda medida:" << endl;
        cin >> m2.incerteza;

        double discrepancia = fabs(m1.valor - m2.valor);
        double incertezaTotal = m1.incerteza + m2.incerteza;

        cout << "Resultado da operacao " << k + 1 << ":" << endl;
        cout << "Discrepancia: " << discrepancia << endl;
        cout << "Incerteza Total: " << incertezaTotal << endl;

        if (discrepancia > incertezaTotal) {
            cout << "Discrepancia significativa" << endl;
        } else {
            cout << "Discrepancia insignificante" << endl;
        }
    }
}
```

Case



O switch case foi usado para selecionar cada função a partir da necessidade do usuário, com ele é possível adicionar várias opções e o que cada uma vai prosseguir

```
int main() {
    int opcao;
    menu();
    cin >> opcao;

    switch (opcao) {
        case 1:
            ErroAleatorio();
            break;
        case 2:
            Soma();
            break;
        case 3:
            Subtracao();
            break;
        case 4:
            Multiplicacao();
            break;
        case 5:
            Divisao();
            break;
        case 6:
            potencia();
            break;
        case 7:
            MultPorEscalar();
            break;
        case 8:
            discrepancia();
            break;
        default:
            cout << "Opcao invalida" << endl;
    }

    return 0;
}
```

github



conclusão



Os experimentos físicos frequentemente geram grandes volumes de dados. A programação permite automatizar o armazenamento e a análise desses dados de forma eficiente e precisa. Ela também pode ser usada para desenvolver cálculos com alta precisão e repetibilidade, minimizando erros humanos e melhorando a confiabilidade dos experimentos. Assim, percebe-se a programação como ferramenta de ampla aplicação e utilidades em diversos meios.



obrigado

