1. *#include* <iostream>
2. *#include* <vector>
3. *using* *namespace* std;
4. *int* main() {
5. *int* n\_alunos;
6. cout *<<* "Insira o número de alunos: ";
7. cin *>>* n\_alunos;
8. vector*<int>* notas(n\_alunos);
9. vector*<int>* histograma(5, 0);
10. *for* (*int* i *=* 0; i *<* n\_alunos; i*++*) {
11. cout *<<* "Insira a nota " *<<* i*+*1 *<<* ": ";
12. cin *>>* notas*[*i*]*;
13. *int* indice *=* (notas*[*i*]* *>=* 9) *?* 0 *:*
14. (notas*[*i*]* *>=* 7) *?* 1 *:*
15. (notas*[*i*]* *>=* 5) *?* 2 *:*
16. (notas*[*i*]* *>=* 3) *?* 3 *:* 4;
17. histograma*[*indice*]++*;
18. }
19. *char* conceitos[5] *=* {'A', 'B', 'C', 'D', 'E'};
20. *for* (*int* i *=* 0; i *<* 5; i*++*) {
21. cout *<<* conceitos[i] *<<* ": ";
22. *for* (*int* j *=* 0; j *<* histograma*[*i*]*; j*++*) {
23. cout *<<* "\*";
24. }
25. cout *<<* endl;
26. }
27. *return* 0;
28. }

2.

*#include* <iostream>

*#include* <cmath>

*using* *namespace* std;

*class* equacaoSegundoGrau {

*private:*

*double* a, b, c;

*public:*

equacaoSegundoGrau (*double* a, *double* b, *double* c) : a(a), b(b), c(c) {}

*double* calcularDelta() {

*return* pow(b, 2) *-* 4 *\** a *\** c;

}

*void* exibirCoeficientes() {

cout *<<* "Coeficientes: a = " *<<* a *<<* ", b = " *<<* b *<<* ", c = " *<<* c *<<* endl;

}

};

*int* main() {

*double* a, b, c;

cout *<<* endl;

cout *<<* "Cálculo de equações do segundo grau em C++" *<<* endl;

cout *<<* endl;

cout *<<* "Digite o coeficiente A: " *<<* endl;

cin *>>* a;

cout *<<* "Digite o coeficiente B: " *<<* endl;

cin *>>* b;

cout *<<* "Digite o coeficiente C: " *<<* endl;

cin *>>* c;

equacaoSegundoGrau equacao(a, b, c);

equacao.exibirCoeficientes();

*double* delta *=* equacao.calcularDelta();

cout *<<* "Δ = " *<<* delta *<<* endl;

*return* 0;

}

3.

*#include* <iostream>

*#include* <vector>

*using* *namespace* std;

*class* vetorNumerosInteiros {

*private:*

vector*<int>* vetor;

*public:*

vetorNumerosInteiros(*int* *tamanho*) : vetor(*tamanho*) {}

*void* getValores() {

cout *<<* "Digite os valores do vetor: \n";

*for* (*int* i *=* 0; i *<* vetor.size(); i*++*) {

cout *<<* "Valor " *<<* i *+* 1 *<<* ": ";

cin *>>* vetor*[*i*]*;

}

}

*void* doubleValores() {

*for* (*int* i *=* 0; i *<* vetor.size(); i*++*) {

vetor*[*i*]* *\*=* 2;

}

}

*void* showValores() *const* {

cout *<<* "Valores dentro do vetor: \n";

*for* (*int* i *=* 0; i *<* vetor.size(); i*++*) {

cout *<<* vetor*[*i*]* *<<* " ";

}

cout *<<* endl;

}

};

*int* main() {

*int* tamanhoVetor;

cout *<<* "Informe a quantidade de itens do vetor: ";

cin *>>* tamanhoVetor;

vetorNumerosInteiros vetorUsuario(tamanhoVetor);

vetorUsuario.getValores();

vetorUsuario.doubleValores();

vetorUsuario.showValores();

*return* 0;

}

4.

*#include* <iostream>

*using* *namespace* std;

*int* recursiveSum(*int* *a*, *int* *b*) {

*if* (a *==* b) {

*return* a;

}

*return* a *+* recursiveSum(a *+* 1, b);

}

*int* main () {

*int* a,b;

cout *<<* "Inisira o valor de A: " *<<* endl;

cin *>>* a;

cout *<<* "Inisira o valor de B: " *<<* endl;

cin *>>* b;

*if* (a *<* b) {

*int* soma *=* recursiveSum(a, b);

cout *<<* "Soma = " *<<* soma *<<* endl;

} *else* {

cout *<<* "Operação inválida. Insira um valor onde B seja maior que a A." *<<* endl;

}

}

5.

*#include* <iostream>

*#include* <cstdlib>

*#include* <ctime>

*#include* <algorithm>

*using* *namespace* std;

*int* bbr(*int* *v*[], *int* *baixo*, *int* *alto*, *int* *chave*) {

*if* (*baixo* *>* *alto*) {

*return* *-*1;

}

*int* meio *=* (*baixo* *+* *alto*) */* 2;

*if* (*v*[meio] *==* *chave*) {

*return* meio;

}

*if* (*v*[meio] *<* *chave*) {

*return* bbr(*v*, meio *+* 1, *alto*, *chave*);

} *else* {

*return* bbr(*v*, *baixo*, meio *-* 1, *chave*);

}

}

*int* main() {

*const* *int* size *=* 50;

*int* v[size];

srand(time(nullptr));

*for* (*int* i *=* 0; i *<* size; i*++*) {

v[i] *=* rand() *%* 101;

}

sort(v, v *+* size);

cout *<<* "Vetor ordenado: ";

*for* (*int* i *=* 0; i *<* size; i*++*) {

cout *<<* v[i] *<<* " ";

}

cout *<<* endl;

*int* chave;

cout *<<* "Digite um valor para realizar a busca binária recursiva: ";

cin *>>* chave;

*int* resultado *=* bbr(v, 0, size *-* 1, chave);

*if* (resultado *!=* *-*1) {

cout *<<* "O valor " *<<* chave *<<* " foi encontrado no índice " *<<* resultado *<<* endl;

} *else* {

cout *<<* "O valor " *<<* chave *<<* " não foi encontrado no vetor." *<<* endl;

}

*return* 0;

}

6.

*#include* <iostream>

*#include* <vector>

*#include* <ctime>

*#include* <algorithm>

*#include* <random>

*using* *namespace* std;

*// Funções de ordenação*

*void* bubbleSort(vector<*int*>*&* *v*) {

*int* n *=* *v*.size();

*for* (*int* i *=* 0; i *<* n *-* 1; i*++*) {

*for* (*int* j *=* 0; j *<* n *-* i *-* 1; j*++*) {

*if* (*v[*j*]* *>* *v[*j *+* 1*]*) {

swap(*v[*j*]*, *v[*j *+* 1*]*);

}

}

}

}

*void* selectionSort(vector<*int*>*&* *v*) {

*int* n *=* *v*.size();

*for* (*int* i *=* 0; i *<* n *-* 1; i*++*) {

*int* minIndex *=* i;

*for* (*int* j *=* i *+* 1; j *<* n; j*++*) {

*if* (*v[*j*]* *<* *v[*minIndex*]*) {

minIndex *=* j;

}

}

swap(*v[*i*]*, *v[*minIndex*]*);

}

}

*void* insertionSort(vector<*int*>*&* *v*) {

*int* n *=* *v*.size();

*for* (*int* i *=* 1; i *<* n; i*++*) {

*int* key *=* *v[*i*]*;

*int* j *=* i *-* 1;

*while* (j *>=* 0 *&&* *v[*j*]* *>* key) {

*v[*j *+* 1*]* *=* *v[*j*]*;

j*--*;

}

*v[*j *+* 1*]* *=* key;

}

}

*void* quicksort(vector<*int*>*&* *v*, *int* *low*, *int* *high*) {

*if* (*low* *<* *high*) {

*int* pivot *=* *v[high]*;

*int* i *=* *low* *-* 1;

*for* (*int* j *=* *low*; j *<* *high*; j*++*) {

*if* (*v[*j*]* *<=* pivot) {

i*++*;

swap(*v[*i*]*, *v[*j*]*);

}

}

swap(*v[*i *+* 1*]*, *v[high]*);

*int* partitionIndex *=* i *+* 1;

quicksort(*v*, *low*, partitionIndex *-* 1);

quicksort(*v*, partitionIndex *+* 1, *high*);

}

}

*void* mergesort(vector<*int*>*&* *v*, *int* *left*, *int* *right*) {

*if* (*left* *<* *right*) {

*int* mid *=* *left* *+* (*right* *-* *left*) */* 2;

mergesort(*v*, *left*, mid);

mergesort(*v*, mid *+* 1, *right*);

*// Mesclando os dois vetores*

*int* n1 *=* mid *-* *left* *+* 1;

*int* n2 *=* *right* *-* mid;

vector*<int>* leftArray(n1), rightArray(n2);

*for* (*int* i *=* 0; i *<* n1; i*++*) {

leftArray*[*i*]* *=* *v[left* *+* i*]*;

}

*for* (*int* j *=* 0; j *<* n2; j*++*) {

rightArray*[*j*]* *=* *v[*mid *+* 1 *+* j*]*;

}

*int* i *=* 0, j *=* 0, k *=* *left*;

*while* (i *<* n1 *&&* j *<* n2) {

*if* (leftArray*[*i*]* *<=* rightArray*[*j*]*) {

*v[*k*]* *=* leftArray*[*i*]*;

i*++*;

} *else* {

*v[*k*]* *=* rightArray*[*j*]*;

j*++*;

}

k*++*;

}

*while* (i *<* n1) {

*v[*k*]* *=* leftArray*[*i*]*;

i*++*;

k*++*;

}

*while* (j *<* n2) {

*v[*k*]* *=* rightArray*[*j*]*;

j*++*;

k*++*;

}

}

}

*int* main() {

*// Inicializa o gerador de números aleatórios*

default\_random\_engine rng(*static\_cast<unsigned>*(time(nullptr)));

uniform\_int\_distribution*<int>* dist(0, 100); *// Valores entre 0 e 100*

*// Cria o vetor de tamanhos*

vector*<int>* tamanhos;

tamanhos.push\_back(100);

tamanhos.push\_back(1000);

tamanhos.push\_back(10000);

tamanhos.push\_back(100000);

*// Loop para cada tamanho de vetor*

*for* (*size\_t* i *=* 0; i *<* tamanhos.size(); i*++*) {

*int* tamanho *=* tamanhos*[*i*]*;

*// Gerar um vetor com valores aleatórios*

vector*<int>* vOriginal(tamanho);

*for* (*int* i *=* 0; i *<* tamanho; i*++*) {

vOriginal*[*i*]* *=* dist*(*rng*)*;

}

cout *<<* "Tamanho do vetor: " *<<* tamanho *<<* endl;

*// Copiar o vetor original para ser ordenado por cada algoritmo*

vector*<int>* v;

*// Bubble Sort*

v *=* vOriginal;

*clock\_t* inicio *=* clock();

bubbleSort(v);

*clock\_t* fim *=* clock();

*double* tempoBubbleSort *=* *double*(fim *-* inicio) */* CLOCKS\_PER\_SEC;

cout *<<* "Tempo de Bubble Sort: " *<<* tempoBubbleSort *<<* " segundos" *<<* endl;

*// Selection Sort*

v *=* vOriginal;

inicio *=* clock();

selectionSort(v);

fim *=* clock();

*double* tempoSelectionSort *=* *double*(fim *-* inicio) */* CLOCKS\_PER\_SEC;

cout *<<* "Tempo de Selection Sort: " *<<* tempoSelectionSort *<<* " segundos" *<<* endl;

*// Insertion Sort*

v *=* vOriginal;

inicio *=* clock();

insertionSort(v);

fim *=* clock();

*double* tempoInsertionSort *=* *double*(fim *-* inicio) */* CLOCKS\_PER\_SEC;

cout *<<* "Tempo de Insertion Sort: " *<<* tempoInsertionSort *<<* " segundos" *<<* endl;

*// Quick Sort*

v *=* vOriginal;

inicio *=* clock();

quicksort(v, 0, tamanho *-* 1);

fim *=* clock();

*double* tempoQuickSort *=* *double*(fim *-* inicio) */* CLOCKS\_PER\_SEC;

cout *<<* "Tempo de Quick Sort: " *<<* tempoQuickSort *<<* " segundos" *<<* endl;

*// Merge Sort*

v *=* vOriginal;

inicio *=* clock();

mergesort(v, 0, tamanho *-* 1);

fim *=* clock();

*double* tempoMergeSort *=* *double*(fim *-* inicio) */* CLOCKS\_PER\_SEC;

cout *<<* "Tempo de Merge Sort: " *<<* tempoMergeSort *<<* " segundos" *<<* endl;

cout *<<* endl; *// Separador entre tamanhos*

}

*return* 0;

}

A screenshot of a computer screen

Description automatically generated

7.

*#include* <iostream>

*#include* <vector>

*#include* <string>

*using* *namespace* std;

*const* *int* MAX\_FUNCIONARIOS *=* 100;

*const* *int* MAX\_DEPARTAMENTOS *=* 8;

*// Estrutura para armazenar as informações de um funcionário*

*struct* Funcionario {

string nome;

*int* idade;

*char* sexo;

*int* tempoDeCasa;

*float* salario;

};

*int* main() {

vector*<*Funcionario*>* funcionarios(MAX\_FUNCIONARIOS);

*int* numFuncionarios;

*// Leitura do número de funcionários e suas informações*

cout *<<* "Digite o número de funcionários: ";

cin *>>* numFuncionarios;

*for* (*int* i *=* 0; i *<* numFuncionarios; i*++*) {

cout *<<* "\nInformações do funcionário " *<<* i *+* 1 *<<* ":\n";

cout *<<* "Nome: ";

cin *>>* funcionarios*[*i*]*.nome;

cout *<<* "Idade: ";

cin *>>* funcionarios*[*i*]*.idade;

cout *<<* "Sexo (M/F): ";

cin *>>* funcionarios*[*i*]*.sexo;

cout *<<* "Tempo de casa (anos): ";

cin *>>* funcionarios*[*i*]*.tempoDeCasa;

cout *<<* "Salário (R$): ";

cin *>>* funcionarios*[*i*]*.salario;

}

*// a) Procurar funcionário pelo nome*

string nomeProcurado;

cout *<<* "\nDigite o nome do funcionário a ser procurado: ";

cin *>>* nomeProcurado;

cout *<<* "\nFuncionários com o nome " *<<* nomeProcurado *<<* ":\n";

*for* (*int* i *=* 0; i *<* numFuncionarios; i*++*) {

*if* (funcionarios*[*i*]*.nome *==* nomeProcurado) {

cout *<<* "Nome: " *<<* funcionarios*[*i*]*.nome *<<* endl;

cout *<<* "Idade: " *<<* funcionarios*[*i*]*.idade *<<* endl;

cout *<<* "Sexo: " *<<* funcionarios*[*i*]*.sexo *<<* endl;

cout *<<* "Tempo de Casa: " *<<* funcionarios*[*i*]*.tempoDeCasa *<<* " anos" *<<* endl;

cout *<<* "Salário: R$ " *<<* funcionarios*[*i*]*.salario *<<* endl;

}

}

*// b) Número de funcionários de um departamento*

*int* departamentoDesejado;

cout *<<* "\nDigite o número do departamento desejado (1 a 8): ";

cin *>>* departamentoDesejado;

*int* contadorDepartamento *=* 0;

*for* (*int* i *=* 0; i *<* numFuncionarios; i*++*) {

*// Supondo que cada funcionário tem um departamento associado (1 a 8)*

*if* (i *%* MAX\_DEPARTAMENTOS *+* 1 *==* departamentoDesejado) {

contadorDepartamento*++*;

}

}

cout *<<* "Número de funcionários no departamento " *<<* departamentoDesejado *<<* ": " *<<* contadorDepartamento *<<* endl;

*// c) Número de funcionárias do sexo feminino*

*int* contadorFeminino *=* 0;

*for* (*int* i *=* 0; i *<* numFuncionarios; i*++*) {

*if* (funcionarios*[*i*]*.sexo *==* 'F' *||* funcionarios*[*i*]*.sexo *==* 'f') {

contadorFeminino*++*;

}

}

cout *<<* "Número de funcionárias do sexo feminino: " *<<* contadorFeminino *<<* endl;

*// d) Funcionários com o menor e maior tempo de casa*

*int* indiceMenorTempo *=* 0;

*int* indiceMaiorTempo *=* 0;

*for* (*int* i *=* 1; i *<* numFuncionarios; i*++*) {

*if* (funcionarios*[*i*]*.tempoDeCasa *<* funcionarios*[*indiceMenorTempo*]*.tempoDeCasa) {

indiceMenorTempo *=* i;

}

*if* (funcionarios*[*i*]*.tempoDeCasa *>* funcionarios*[*indiceMaiorTempo*]*.tempoDeCasa) {

indiceMaiorTempo *=* i;

}

}

cout *<<* "\nFuncionário com o menor tempo de casa:\n";

cout *<<* "Nome: " *<<* funcionarios*[*indiceMenorTempo*]*.nome *<<* endl;

cout *<<* "Tempo de Casa: " *<<* funcionarios*[*indiceMenorTempo*]*.tempoDeCasa *<<* " anos" *<<* endl;

cout *<<* "\nFuncionário com o maior tempo de casa:\n";

cout *<<* "Nome: " *<<* funcionarios*[*indiceMaiorTempo*]*.nome *<<* endl;

cout *<<* "Tempo de Casa: " *<<* funcionarios*[*indiceMaiorTempo*]*.tempoDeCasa *<<* " anos" *<<* endl;

*// e) Salário médio por departamento*

vector*<float>* salariosDepartamento(MAX\_DEPARTAMENTOS, 0.0);

vector*<int>* contadoresDepartamento(MAX\_DEPARTAMENTOS, 0);

*for* (*int* i *=* 0; i *<* numFuncionarios; i*++*) {

*int* departamento *=* i *%* MAX\_DEPARTAMENTOS;

salariosDepartamento*[*departamento*]* *+=* funcionarios*[*i*]*.salario;

contadoresDepartamento*[*departamento*]++*;

}

cout *<<* "\nSalário médio por departamento:\n";

*for* (*int* i *=* 0; i *<* MAX\_DEPARTAMENTOS; i*++*) {

*if* (contadoresDepartamento*[*i*]* *>* 0) {

cout *<<* "Departamento " *<<* i *+* 1 *<<* ": R$ " *<<* salariosDepartamento*[*i*]* */* contadoresDepartamento*[*i*]* *<<* endl;

}

}

*// f) Idade média por departamento*

vector*<int>* idadesDepartamento(MAX\_DEPARTAMENTOS, 0);

*for* (*int* i *=* 0; i *<* numFuncionarios; i*++*) {

*int* departamento *=* i *%* MAX\_DEPARTAMENTOS;

idadesDepartamento*[*departamento*]* *+=* funcionarios*[*i*]*.idade;

}

cout *<<* "\nIdade média por departamento:\n";

*for* (*int* i *=* 0; i *<* MAX\_DEPARTAMENTOS; i*++*) {

*if* (contadoresDepartamento*[*i*]* *>* 0) {

cout *<<* "Departamento " *<<* i *+* 1 *<<* ": " *<<* *static\_cast<float>*(idadesDepartamento*[*i*]*) */* contadoresDepartamento*[*i*]* *<<* " anos" *<<* endl;

}

}

*// g) Número de funcionários do sexo masculino e feminino, com idade entre 29 e 35 anos*

*int* contadorMasculinoIdade *=* 0;

*int* contadorFemininoIdade *=* 0;

*for* (*int* i *=* 0; i *<* numFuncionarios; i*++*) {

*if* ((funcionarios*[*i*]*.sexo *==* 'M' *||* funcionarios*[*i*]*.sexo *==* 'm') *&&* (funcionarios*[*i*]*.idade *>=* 29 *&&* funcionarios*[*i*]*.idade *<=* 35)) {

contadorMasculinoIdade*++*;

} *else* *if* ((funcionarios*[*i*]*.sexo *==* 'F' *||* funcionarios*[*i*]*.sexo *==* 'f') *&&* (funcionarios*[*i*]*.idade *>=* 29 *&&* funcionarios*[*i*]*.idade *<=* 35)) {

contadorFemininoIdade*++*;

}

}

cout *<<* "\nNúmero de funcionários do sexo masculino (29-35 anos): " *<<* contadorMasculinoIdade *<<* endl;

cout *<<* "Número de funcionários do sexo feminino (29-35 anos): " *<<* contadorFemininoIdade *<<* endl;

*return* 0;

}