Exercício de Programação Estruturada 5

Exercício de Programação Estruturada 5

1.

```c
#include <stdio.h>

void fillGrades(float grades[], int size)
{
    int i;
    for (i = 0; i < size; i++)
    {
        printf("Digite a nota do aluno %d: ", i + 1);
        scanf(" %f", &grades[i]);
    }
}

void calculateAvrg(float firstGrades[], float secondGrades[], float
averages[], int size)
{
    int i;
    for (i = 0; i < size; i++)
    {
        averages[i] = (firstGrades[i] * 2 + secondGrades[i] * 3)/ 5;
    }

}

void printVector(float vet[], int size)
{
    int i;
    for (i = 0; i < size; i++)
    {
        printf("%.2f\n", vet[i]);
    }
}

void gradeStudents(float averages[], int situations[], int size)
{
    /*situações = 1 - foi aprovado direto, 2 - está em recuperação ou 3 -foi
```

```
reprovado direto*/
    int i;
    for (i = 0; i < size; i++)
    {
        if (averages[i] >= 7.0)
        {
            situations[i] = 1;
        }
        else if (averages[i] >= 3.0)
        {
            situations[i] = 2;
        }
        else
        {
            situations[i] = 3;
        }
    }

}

void calculatePerc (int situations[], float* percOfApproved, float*
percOfSummerSchool, float* percOfFailed, float sizeF)
{
    int i;
    float approved, summerSchool, failed;
    approved = 0;
    summerSchool = 0;
    failed = 0;
    for (i = 0; i < sizeF; i++)
    {
        switch (situations[i])
        {
        case 1:
            approved = approved + 1;
            break;
        case 2:
            summerSchool = summerSchool + 1;
            break;
        case 3:
            failed = failed + 1;
            break;
        }
    }
```

```
    *percOfApproved = approved * 100 / sizeF;
    *percOfSummerSchool = summerSchool * 100 / sizeF;
    *percOfFailed = failed * 100 / sizeF;
}

float findSmallest(float averages[], int size)
{
    int i;
    float smallest;

    smallest = 0.0;
    for (i = 0; i < size; i++)
    {
        if (averages[i] < smallest)
        {
            smallest = averages[i];
        }
    }
    return smallest;
}

float findGreatest(float averages[], int size)
{
    int i;
    float greatest;

    greatest = 10.0;
    for (i = 0; i > size; i++)
    {
        if (averages[i] < greatest)
        {
            greatest = averages[i];
        }
    }
    return greatest;
}

float calculateClassAverage(float averages[], int size, float sizeF)
{
    float sum;
    int i;

    for (i = 0; i < size; i++)
```

```
    {
       sum = sum + averages[i];
    }
    return sum / sizeF;
}

int calculateUnderAverage(float averages[], float classAvrg, int size)
{
    int i, studentsUnder;
    studentsUnder = 0;
    for (i = 0; i < size; i++)
    {
       if (averages[i] < classAvrg)
       {
            studentsUnder = studentsUnder + 1;
       }

    }
    return studentsUnder;
}

int main()
{
    int size = 5;
    float sizeF = 5;
    int i, situations[size], UnderAvrg;
    float firstGrades[size], secondGrades[size], averages[size],
percOfApproved, percOfSummerSchool, percOfFailed, greatestAvrg,
smallestAvrg, classAvrg;

    printf("Preenchendo as notas da primeira avaliação:\n");
    fillGrades(firstGrades, size);
    printf("Preenchendo as notas da segunda avaliação:\n");
    fillGrades(secondGrades, size);
    calculateAvrg(firstGrades, secondGrades, averages, size);
    printf("Notas da primeira avaliação: \n");
    printVector(firstGrades, size);
    printf("Notas da segunda avaliação: \n");
    printVector(secondGrades, size);
    printf("Médias: \n");
    printVector(averages, size);
    gradeStudents(averages, situations, size);
    calculatePerc (situations, &percOfApproved, &percOfSummerSchool,
```

```
&percOfFailed, sizeF);
    printf("%.2f%% dos alunos foram aprovados direto, %.2f%% dos alunos foram para a recuperação e %.2f%% dos alunos foram reprovados.\n", percOfApproved, percOfSummerSchool, percOfFailed);
    smallestAvrg = findSmallest(averages, size);
    greatestAvrg = findGreatest(averages, size);
    printf("A menor média foi %.2f%% e a maior média foi %.2f%%.\n", smallestAvrg, greatestAvrg);
    classAvrg = calculateClassAverage(averages, size, sizeF);
    UnderAvrg = calculateUnderAverage(averages, classAvrg, size);
    printf("A média de notas da sala foi %.2f e %d alunos estão abaixo dela.\n", classAvrg, UnderAvrg);
    return 0;
}
```