# 1. Projekt

## Kitűzött feladatok:

### 1. Csapatok kialakítása

Válasszatok csapattársat a következő táblázat alapján (https://docs.google.com/spreadsheets/d/1zt0Y2GWmNfH7TQAEfnPltOCrVvwZl677cDi631-SA_o/edit?usp=sharing). Mindenki csak a másik oszlopból választhat csapattársat. A csapatok összetételét email-ben küldje el az egyik csapattag legkésőbb március 21. vasárnap éjfélig a következő címre (palmarozalia.osztian@gmail.com).

### 2. Téma kiválasztása

Válasszatok közösen egy, a mintafeladathoz hasonló témát. A témát a következő táblázatba írjátok be:

(https://docs.google.com/spreadsheets/d/1zt0Y2GWmNfH7TQAEfnPltOCrVvwZl677cDi631-SA_o/edit?usp=sharing) legkésőbb március 23. szerda este 10-ig a csapatotoknak megfelelő sorba. Minden csapat témája különböző kell legyen.

### 3. Projekt előkészítése

A csapat egyik tagja készítsen egy GitHub repository-t és hívja meg a csapattársát és a gyakorlatvezető tanárt (palmarozaliaosztian) collaborator-nak.

Határidő: március 25. csütörtök éjfél.

### 4. Projekt megvalósítása

A projekt véglegesítésére 3 hét áll rendelkezésre. Utolsó commit: április 14. szerda éjfél. Mindkét csapattárs egyenlő mértékben kell kivegye a munkából a részét.

# 1. Projekt

## Megkötések:

- A mintafeladathoz hasonlóan legalább két különböző típusdefiníciót kell tartalmazzon a projekt (pl: Student és Classroom), melyek között kell legyen valamilyen kapcsolat (pl. tartalmazási: minden osztályteremhez tartoznak diákok).
- Mindkét (vagy több) típushoz tartozzon legalább 15-15 függvény/eljárás. Ha valaki több típust használ ez a mennyiség csökkenhet, a lényeg, hogy a projekt összességében tekintve rendelkezzen legalább 30 funkcionalitással.
- Legyen egy jelentős adathalmaz biztosítva (min. 100 adat) a teszteléshez. Ez lehet valós, vagy generált adat.
- A main biztosítson lehetőséget az adathalmazzal való tesztelésre (ha ezt az opciót választja a felhasználó a megfelelő adatokkal feltölti a program az adott változókat), de egy interaktív menü-vel való működésre is (lehessen a rendelkezésre álló funkcionalitásokat tesztelni úgy, hogy a program minden alkalommal megkérdezi a felhasználót mit szeretne).
- A két hét alatti időszak során szükséges bemutatni az addigi haladást és fejlődést a projektet illetően. Lehet feltenni kérdéseket, lehet segítséget kérni az implementáció és a projekt minőségét illetően.

## Mintafeladat:

- Adottak a Student és Classroom típusdefiníciók és a hozzájuk tartozó header fájlok. A mintafeladat egy adott diákra és a teljes osztályteremre (diákok összessége) vonatkozó funkcionalitások közül szemléltet néhányat.

```
typedef struct Student {
    char* name;
    char classroom;
    int grade;
    bool gender;
    double* marks;
    int numberOfMarks;
}Student;

Student* create(int maxNumberOfMarks);

void destroy(Student* student);

//Student kiiratása a képernyõre
void print(Student* student);

//Student kiiratása állományba
void printToFile(FILE* file, Student* student);
//Adott Student jegyei átlagának kiszámítása
double averageOfMarks(Student* student);
```

# 1. Projekt

```
//ClassRoom struktúra definiálása
typedef struct ClassRoom {
    Student** classRoom;
    int numberOfStudents;
}ClassRoom;

//ClassRoom típus létrehozása
ClassRoom* createClass(int numberOfStudents);

//ClassRoom adatainak beolvasása
ClassRoom* readStudentsData(const char* fileName);

//ClassRoom adatainak kiiratása a képernyõre
void printClass(ClassRoom* classRoom);

//ClassRoom típus felszabadítása
void destroyClass(ClassRoom* classRoom);

//ClassRoom adatainak kiiratása állományba
void printClassToFile(const char* fileName, ClassRoom* classRoom);

//Adott ClassRoom - ban található diákok átlagának kiszámítása
(osztályátlag)
double averageOfClass(ClassRoom* classRoom);

//Nemek eloszlása (kik vannak többen)
void genderProportion(ClassRoom* classRoom);

//Egy adott névvel rendelkezõ diák adatainak kiiratása
void detailsOfSpecificStudent(ClassRoom* classRoom, char* name);

//ClassRoom tanulóinak névsor szerinti rendezése
void sortClassByName(ClassRoom* classRoom);

//ClassRoom tanulóinak évfolyam és azon belül névsor szerinti rendezése
void sortClassByGradeAndName(ClassRoom* classRoom);
```

```
MAIN

Tesztelés, adathalmazból illetve interaktív menüvel

int main() {

    int option;
    bool repeat = true;


    while(repeat) {
        printf("Lehetseges muveletek:\n"
               "0. Kilepes\n"
               "1. Osztaly letrehozasa\n"
               "2. Diakok adatainak beolvasasa fajlbol\n"
               "3. Diakok kiiratasa a kepernyore\n"
               "4. Diakok kiiratasa allomanyba\n"
               "5. Osztalyatlag kiszamitasa\n"
               "6. Diakok rendezese nev szerint\n"
               "stb.....");
```

# 1. Projekt

```
        printf("Valassz opciot: ");
        scanf("%i", &option);

        switch(option) {
            case 0:
                printf("Kilepes...");
                repeat = false;
                break;
            case 1:
                ///createClass megfelelo meghivasa
                break;
            case 2:
                ///readStudentsData megfelelo meghivasa
                break;
            case 3:
                ///printClass megfelelo meghivasa
                break;
            case 4:
                ///printClassToFile megfelelo meghivasa
                break;
            case 5:
                ///averageOfClass megfelelo meghivasa
                break;
            case 6:
                ///sortClassByName megfelelo meghivasa
                break;
            default:
                printf("Helytelen opcio. Probald ujra");
                break;
        }

    }

    ///szukseges felszabaditasok elvegzese



    return 0;
}
```