Навчальна практика з програмування

Мета навчальної практики з програмування: поглиблення і закріплення здобутих за час навчання теоретичних і практичних знань, умінь та навичків з навчальної дисципліни "Алгоритмічних мови та програмування".

Під час практики студенти під керівництвом викладача закріплюють набуті навички програмування та опрацьовують різноманітні теми з програмування.

Орієнтовний регламент навчального дня практики:

- обговорення з керівником практики поточної теми;
- індивідуальне опрацювання комплексних задач;
- документальне оформлення ходу розв’язання найскладнішої проблеми.

Під час проходження практики студенти щодня документально оформлюють (звіт) виконання найскладніших та/або найцікавіших фрагментів отриманих практичних завдань з програмування. При цьому звіт має містити опис завдання, деталізацію інформації стосовно реалізації програми, опис та обґрунтування використання змінних, опис формалізації даної задачі та кроки виконання програми.

Під час проходження студентами навчальної практики керівник практики контролює роботу студентів, виконання поставлених завдань та їх якість, проводить консультації зі студентами під час виконання завдань з метою перевірки роботи та надання допомоги.

Перелік тем, які опрацьовуються під час практики:

- Тип struct. Функції. Робота з файлами.
- Класи, основи об’єктно-орієнтованого програмування.
- Базові алгоритми: сумування, пошук, вирішення, підрахунок, знаходження мінімуму/максимуму.
- Розв’язування різноманітних задач про вибірки.
- Знаходження мінімуму та максимуму серед елементів із даною властивістю.
- Алгоритм сортування.
- Пошук k-тового найбільшого та найменшого елементу.
- Задачі на багаторівневі впорядкування.
- Об`єднання та перерізи. Методи програмування задач на об`єднання впорядкованих множин.

Приклад комплексної задачі, яка опрацьовується студентами під час практики:

Завдання «Вимірювання»

Є файл *meresek.txt*, який містить дані вимірювань експерименту. У файлі може бути максимум 300 вимірювань. Кожен рядок містить одне вимірювання з його ідентифікатором, типом вимірювання (від 0 до 4), значеннями F і Z вимірювання.

Наприклад:

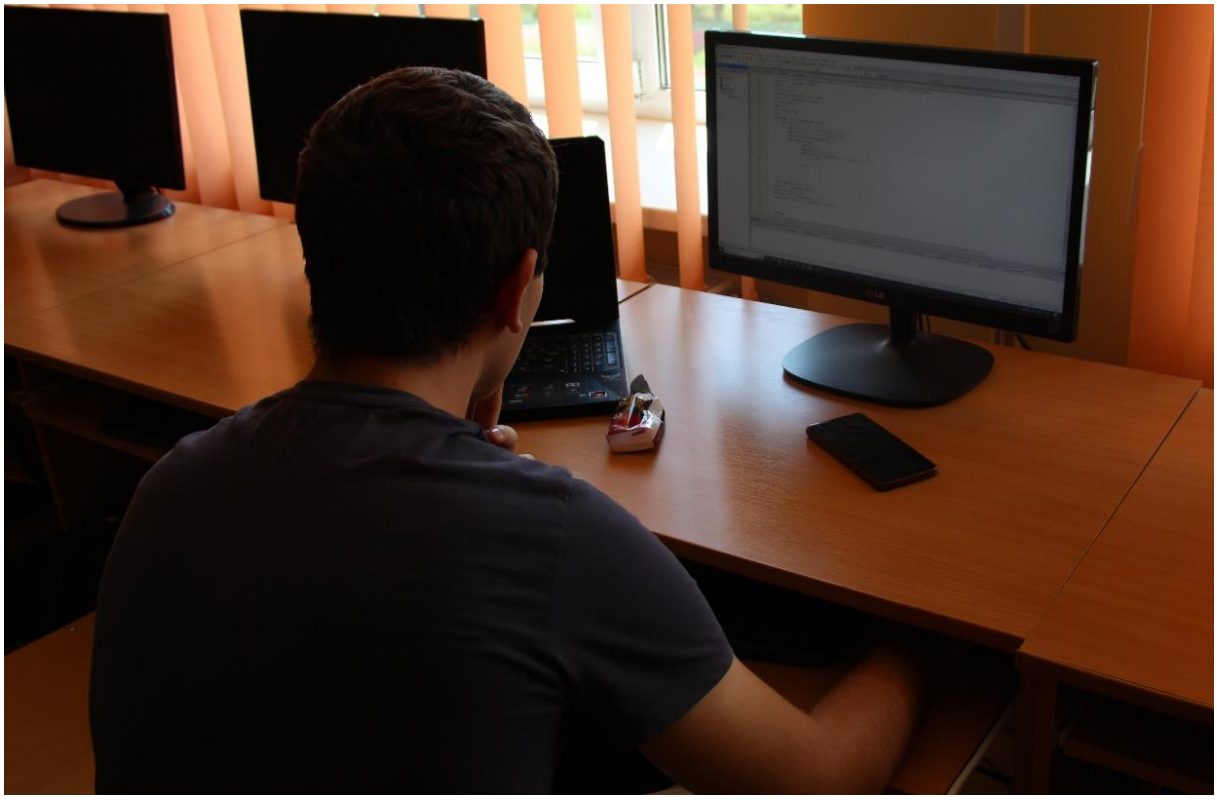
```
R1458 1 24.76 7.25
R1459 1 21.77 10.78
R1460 2 49.21 24.14
R1461 0 11.58 5.74
R1462 2 54.64 27.21
```

У першому рядку вищезазначеного прикладу знаходиться вимірювання з ідентифікатором `R1458`, типом `1`. Значення `F` цього вимірювання становить `24.76`, а значення `Z` - `7.25`.

Напишіть програму, яка, використовуючи дані файлу *meresek.txt*, відповідає на наступні питання! Збережіть вихідний код програми під назвою `meresek`! Перед виведенням на екран результатів завдань, що вимагають виведення на екран, виведіть номер завдання (наприклад: `Завдання 3`). Якщо ви запитуєте дані у користувача, покажіть на екрані, яке значення ви очікуєте.

1. Прочитайте дані з файлу *meresek.txt*. Виведіть, скільки вимірювань містить файл!
2. Визначте, скільки вимірювань було проведено з кожного типу!
3. Визначте, чи вимірювання з найбільшими значеннями `F` та `Z` були проведені в одному й тому ж вимірюванні!
4. Визначте, скільки значень `F` перевищили середнє!
5. Відомо, що будь-яке значення `F` вимірювання не може перевищувати 100. Створіть файли, в яких зберете вимірювання з діапазонів 0-25, 25-50, 50-75 та 75-100! Назви файлів мають бути діапазонами вимірювань. Перший рядок файлу містить кількість вимірювань, що в ньому знаходяться.
6. Значенню вимірювання, яке часто вимірюється у певному типі, приділяється важлива роль. Визначте серед значень Z для кожного типу, які з них є такими значеннями!

Світлини з навчальної практики з програмування
2022-2023



Програмні коди із виконаних звітів робіт студентів:

```cpp
struct Vetelek{
    int nap, ramator;
    string uzenet;
};
...
Vetelek vetel[220];
...
int napsz, sorsz;
cout<<"Adja meg a nap sorszamat(1-11): "; cin>>napsz;
cout<<"Adja meg a radioamator sorszamat(1-20): "; cin>>sorsz;
int sum, perpos=0, elsoszok=0;
string felnott, kolyok, uzenet;

for(int i=0;i<elemszam;i++){
    if(vetel[i].nap==napsz && vetel[i].ramator==sorsz){
        uzenet=vetel[i].uzenet;
        break;
    }
}
if(uzenet==""){
    cout<<"nincs ilyen feljegyzes"<<endl;
}else{
    perpos=uzenet.find('/');
    elsoszok=uzenet.find(' ');
    if(perpos<uzenet.length() && elsoszok<uzenet.length()){
        felnott=uzenet.substr(0,perpos);
        kolyok=uzenet.substr(perpos+1,elsoszok-perpos-1);
        cout<<felnott<<","<<kolyok<<endl;
        if(szame(felnott) && szame(kolyok)){
            sum=stoi(felnott)+stoi(kolyok);
            cout<<"A megfigyelt egyedek szama: "<<sum<<endl;
        }else{
            cout<<"Nincs informacio"<<endl;
        }
    }else{
        cout<<"Nincs informacio"<<endl;
    }
}
```

```cpp
struct Hianyzas{
    string vnev, knev;
    string orak;
};
struct Nap
{
    int honap, nap, darab;
    Hianyzas hianyzas[50];
};
struct Tanulok
{
    string nevek;
    int hianyzasok;
};
...
Nap nap[600];
...
Tanulok tanulok[50];
bool benne = 0;
int k = 0;
for (int i=0; i<db; i++){
    for (int j=0; j<nap[i].darab; j++){
        benne = 0;
        string nev = nap[i].hianyzas[j].vnev + " " + nap[i].hianyzas[j].knev;
        for (int l=0; l<k; l++){
            if (nev == tanulok[l].nevek){
                benne = 1;
            }
        }
        if (!benne){
            tanulok[k].nevek = nev;
            k++;
        }
    }
}
```

```cpp
for (int i=0; i<db; i++){
    for (int j=0; j<nap[i].darab; j++){
        string nev = nap[i].hianyzas[j].vnev + " " + nap[i].hianyzas[j].knev;
        for (int l=0; l<k; l++){
            if (nev == tanulok[l].nevek){
                for (int h=0; h<nap[i].hianyzas[j].orak.length(); h++){
                    if (nap[i].hianyzas[j].orak[h] == 'X' || nap[i].hianyzas[j].orak[h] == 'I')
                        tanulok[l].hianyzasok++;
                }
            }
        }
    }
}
for (int i=0; i<k-1; i++){
    for (int j=i+1; j<k; j++)
        if (tanulok[i].hianyzasok < tanulok[j].hianyzasok){
            swap(tanulok[i], tanulok[j]);
        }
}

int i=0;
cout << "A legtobbet hianyzo tanulok: ";
do{
    cout << tanulok[i].nevek << " ";
    i++;
}
while (tanulok[i].hianyzasok == tanulok[i-1].hianyzasok);
cout << endl;
```

```cpp
ofstream g("adaas.txt");
string fejt[11];
for (int i=0; i<11; i++){
    fejt[i]=t[index[i][0]].text;
}
for (int i=0; i<11; i++){
    for (int j=1; j<tomb[i]; j++){
        for(int k=0; k<90;k++){
            if(t[index[i][j]].text[k]!='#' && t[index[i][j]].text[k]!='$' && fejt[i][k]=='#'){
                fejt[i][k]=t[index[i][j]].text[k];
            }
        }
    }
}
for (int i=0; i<11; i++){
    g<<i+1<<":"<<endl<<fejt[i]<<endl<<endl;
}
g.close();
```

```cpp
#include <iostream>
#include <fstream>

using namespace std;
int loertek (string s){
int ossz=0, pont =20;
int i=0;
while(i<s.length()){
    if(s[i]!='-'){
        ossz+=pont;
    }
    else{
        if(pont>0){
            pont--;
        }
    }
    i++;
}
return ossz;
}
int main()
{
    ifstream f("verseny.txt");
    int elemszam;
    f>>elemszam;
    string t[elemszam];
    int db=0;
    f.get();
    while(!f.eof() && db<elemszam){
        getline(f,t[db++]);
    }
    for(int i=0; i<elemszam;i++){
        cout<<i+1<<" "<<t[i]<<endl;
    }
```

```cpp
    cout<<"2. feladat"<<endl;
    for(int i=0; i<elemszam;i++){
        bool jo=false;
        for(int j=0; j<t[i].length();j++){
            if(t[i][j]==t[i][j+1] && t[i][j]=='+'){
                jo=true;
                break;
            }
        }
        if(jo){
            cout<<i+1<<" ";
        }
    }
    cout<<endl<<"3. feladat"<<endl;
    int maxx=0;
    for(int i=0; i<elemszam;i++){
        if(t[i].length()>t[maxx].length()){
            maxx=i;
        }
    }
    cout<<maxx+1;
    cout<<endl<<"4. feladat"<<endl;
    cout<<loertek(t[3]);
    cout<<endl<<"5. feladat"<<endl;
    int a;
    cin>>a;
    db=0;
    int tomb[40];
    for(int i=0; i<t[a-1].length();i++){
        if(t[a-1][i]=='+'){
            tomb[db]=i;
            db++;
        }
    }
```

```cpp
    int utan=0;
    int ind=0;
    for(int i=0; i<=db;i++){
        int seged=1;
        int seged1=0;
        for(int j=i; j<=db;j++){
            if(tomb[i]==tomb[j]-seged){
                seged1++;
                seged++;
            }
        }
        if(seged1>utan){
            utan =seged1;
            ind=i;
        }
    }
    cout<<"a bekert sorszamu versenyzo a kovetkezoket lotte el: ";
    for(int i=0; i<db;i++){
        cout<<tomb[i]+1<<" ";
    }
    cout<<endl<<db<<" kororngot talalt el"<<endl;
    cout<<utan+1<<" hosszu lovest talalt el egymas utan "<<endl;
    cout<<loertek(t[a-1])<<" pontot ert el"<<endl;
    ofstream g("sorrend.txt");
    cout<<"6. feladat"<<endl;
    int adat[2][elemszam];
    for(int i=0; i<elemszam;i++){
        adat[0][i]=i;
        adat[1][i]=loertek(t[i]);
    }
    int kezd=0;
    kezd=0;
```

```cpp
    kezd=0;
    while(kezd<elemszam){
        ind=kezd;
        for(int i=kezd; i<elemszam;i++){
            if(adat[1][i]>adat[1][ind]){
                ind=i;
            }
        }
        swap(adat[0][kezd],adat[0][ind]);
        swap(adat[1][kezd],adat[1][ind]);
        kezd++;
    }
    int hely=1;
    for(int i=0; i<elemszam;i++){
        g<<hely<<" "<<adat[0][i]+1<<" "<<adat[1][i]<<endl;
        if(adat[1][i]!=adat[1][i+1]){
            hely++;
        }
    }
    g.close();
    return 0;
}
```

```cpp
#include <iostream>
#include <fstream>

using namespace std;

struct Jarmu
{
    int ora, perc, mp;
    string rendszam;
    string alakit(){
        string oras, perces, mps;
        if (ora < 10)
            oras = '0' + to_string(ora) + ':';
        else
            oras = to_string(ora) + ':';

        if (perc < 10)
            perces = '0' + to_string(perc) + ':';
        else
            perces = to_string(perc) + ':';

        if (mp < 10)
            mps = '0' + to_string(mp);
        else
            mps = to_string(mp);
        return oras + perces + mps;
    }
};

int mpbe (int ora, int perc, int mp)
{
    return ora*3600+perc*60+mp;
}
```

```cpp
int main()
{
    Jarmu jarmu[1000];
    int db = 0;
    ifstream f("jarmu.txt");
    while (!f.eof()){
        f >> jarmu[db].ora >> jarmu[db].perc >> jarmu[db].mp >> jarmu[db].rendszam;
        db++;
    }
    db--;
    f.close();

    ///2. feladat
    cout << "2. feladat" << endl;
    if (jarmu[db-1].perc != 59 && jarmu[db-1].mp != 59)
        cout << "Az ellenorzes napjan minimum " << jarmu[db-1].ora + 1 - jarmu[0].ora << " orat dolgoztak " << endl;
    else
        cout << "Az ellenorzes napjan minimum " << jarmu[db-1].ora - jarmu[0].ora << " orat dolgoztak " << endl;

    ///3. feladat
    cout << endl << "3. feladat" << endl;
    string ora[24];
    for (int i=0; i<24; i++){
        for (int j=0; j<db; j++){
            if (i == jarmu[j].ora){
                ora[i] = jarmu[j].rendszam;
                break;
            }
        }
    }
    for (int i=0; i<24; i++){
        if (ora[i] != "")
            cout << i << ". ora: " << ora[i] << endl;
    }
```

```cpp
    ///4. feladat
    int B = 0, K = 0, M = 0, E = 0; // e- egyeb
    for (int i=0; i<db; i++){
        if (jarmu[i].rendszam[0] == 'B')
            B++;
        else if (jarmu[i].rendszam[0] == 'K')
            K++;
        else if (jarmu[i].rendszam[0] == 'M')
            M++;
        else
            E++;
    }
    cout << endl << "4. feladat" << endl;
    cout << "Busz: " << B << endl;
    cout << "Kamion: " << K << endl;
    cout << "Motor: " << M << endl;
    cout << "Szemelygepkocsi: " << E << endl;

    ///5. feladat
    int maxx = 0;
    string maxkezdet, maxvege;
    for (int i=0; i<db-1; i++){
        int forgalommentes = mpbe(jarmu[i+1].ora, jarmu[i+1].perc, jarmu[i+1].mp) - mpbe(jarmu[i].ora, jarmu[i].perc, jarmu[i].mp);
        if (forgalommentes > maxx){
            maxx = forgalommentes;
            maxkezdet = to_string(jarmu[i].ora) + ':' + to_string(jarmu[i].perc) + ':' + to_string(jarmu[i].mp);
            maxvege = to_string(jarmu[i+1].ora) + ':' + to_string(jarmu[i+1].perc) + ':' + to_string(jarmu[i+1].mp);
        }
    }
    cout << endl << "5. feladat" << endl;
    cout << "Leghosszabb forgalommentes idoszak: " << maxkezdet << "-" << maxvege << endl;
```

```cpp
    ///6. feladat
    string keresett; // = "L**73**";
    cout << endl << "6. feladat" << endl;
    cout << "Adja meg a keresett rendszamot: "; cin >> keresett;
    bool megfelel = 1;
    for (int i=0; i<db; i++){
        megfelel = 1;;
        for (int j=0; j<7; j++){
            if (keresett[j] != '*' && keresett[j] != jarmu[i].rendszam[j]){
                megfelel = 0;
                break;
            }
        }
        if(megfelel)
            cout << jarmu[i].rendszam << endl;
    }

    ///7. feladat
    ofstream f1("vizsgalt.txt");
    Jarmu seged = jarmu[0];
    f1 << jarmu[0].alakit() << " " << jarmu[0].rendszam << endl;
    for (int i=0; i<db; i++){
        while (mpbe(jarmu[i].ora, jarmu[i].perc, jarmu[i].mp)
               - mpbe(seged.ora, seged.perc, seged.mp) < 5*60){
            i++;
        }
        seged = jarmu[i];
        f1 << jarmu[i].alakit() << " " << jarmu[i].rendszam << endl;
    }
    f1.close();

    return 0;
}
```