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;
- розробка поточної теми, створення короткого підсумкового звіту.

Під час практики студенти обговорюють тему дня з керівником практики, потім самостійно готують синопсис теми, який документують у звіті. Після цього працюють над різноманітними комплексними практичними завданнями з програмування.

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:

- Знаходження мінімуму та максимуму серед елементів із даною властивістю.
- Метод простого впорядкування.
- Методи програмування задач на об`єднання впорядкованих множин.
- Задачі на багаторівневі впорядкування.
- Типові задачі на вибір, методи їх розв`язування.
- Пошук k-тового найбільшого та найменшого елементів.

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:

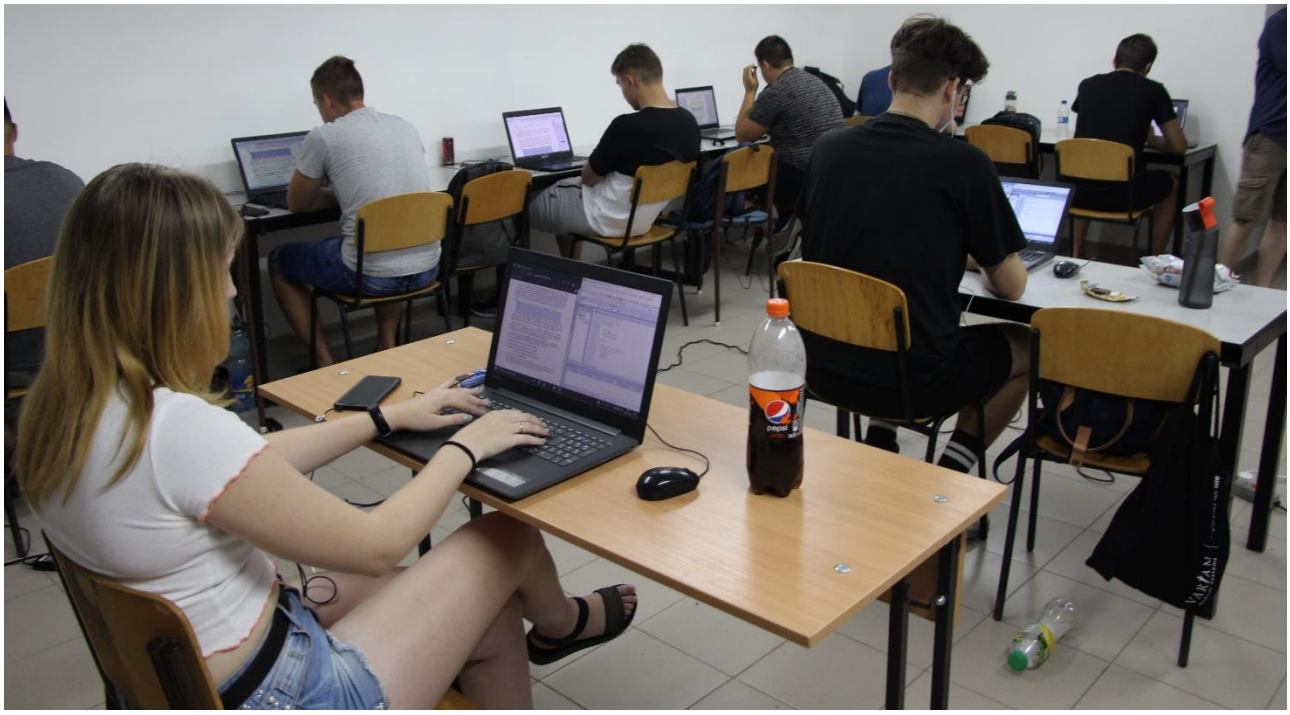
```
R1458 1 24.76 7.25
R1459 1 21.77 10.78
R1460 2 49.21 24.14
R1461 0 11.58 5.74
R1462 2 54.64 27.21
```

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
2020-2021


pepsi

Програмні коди із виконаних звітів робіт студентів:

```
struct Orszag{
    string onev, vnev;
};
...
Orszag t[100];
...
for(int i = db - 1; i > 0; i--){
    for(int j = 0; j < i; j++){
        if(nevek[j].onev > nevek[j+1].onev){
            Orszagok csere = nevek[j];
            nevek[j] = nevek[j+1];
            nevek[j+1] = csere;
        }
    }
}
for(int i = db - 1; i > 0; i--){
    for(int j = 0; j < i; j++){
        if(nevek[j].onev == nevek[j+1].onev &&
                nevek[j].vnev < nevek[j+1].vnev ){
            Orszagok csere;
            csere = nevek[j];
            nevek[j] = nevek[j+1];
            nevek[j+1] = csere;
        }
    }
}
```

```
int x[20], y[20];
...
int z[40];
int i=0, j=0, db=0;
while(i < n && j < m){
    if(x[i] < y[j]){
        z[db] = x[i];
        i++;
    }else if(x[i] == y[j]){
        z[db] = x[i];
        i++;
        j++;
    }else if(x[i] > y[j]){
        z[db] = y[j];
        j++;
    }
    db++;
}
while(i < n){
    z[db] = x[i];
    i++;
    db++;
}
while(j < m){
    z[db] = y[j];
    j++;
    db++;
}
```

```
const int n = 100;
int t2[n];
...
for(int i = 0; i < n; i++){
    t2[i] = t[i];
}
int min_tol_kisebb = t2[0];
for(int i = 0; i < n; i++){
    if(t2[i] < min_tol_kisebb){
        min_tol_kisebb = t2[i];
    }
}
min_tol_kisebb--;
int k_max;
for(int j = 0; j < k; j++){
    k_max = t2[0];
    int k_ind = 0;
    for(int i = 0; i < n; i++){
        if(t2[i] > k_max){
            k_max = t2[i];
            k_ind = i;
        }
    }
    t2[k_ind] = min_tol_kisebb;
}
```

```
CIKLUS I = E+1 -> N
   J = I-1
   AMÍG J > 0 && T[J] > T[J+1]
      T[J], T[J+1] CSERE
      J--
VÉGE
E - az első valahány rendezett elem elemszáma
N - az összes elem elemszáma

int t[100];
...
for (int i=2; i<100; i++){
   int j=i-1;
   while(j>0 && t[j]>t[j+1]){
      int csere = t[j];
      t[j] = t[j+1];
      t[j+1] = csere;
      j--;
   }
}
```

```
const int n = 100;
int t[n];
...
int j = 0;
while(j < n && t[j] % 5 != 0){
    j++;
}
if(j < n){
    int max = t[j];
    for(int i = j+1; i < n; i++){
        if(t[i] % 5 == 0 && t[i] > max){
            max = t[i];
        }
    }
    cout << "A legnagyobb 5-el oszthato szam: " << max << endl;
}else{
   cout << "A tombben nincs 5-el oszthato elem" << endl;
}
```

```
struct Elemek{
   int szam;
   int darab=0;
};
...
int tomb[15];
...
Elemek tomb2[15];
int db = 0;
for(int i=0;i<15;i++){
   marvan=false;
   for(int j=0;j<db;j++){
      if(tomb[i]==tomb2[j].szam){
         tomb2[j].darab++;
         marvan=true;
         break;
      }
   }
   if(marvan==false){
      tomb2[db].szam=tomb[i];
      tomb2[db].darab=1;
      db++;
   }
   marvan=true;
}
```

```
struct Elemek{
   int szam;
   int darab=0;
};
...
int t[15];
...
Elemek elemek[15];
int db = 0;
for (int i=14; i>0; i--) {
   for (int j=0; j<i; j++) {
      if (t[j]>t[j+1]) {
         int csere=t[j];
         t[j]=t[j+1];
         t[j+1]=csere;
      }
   }
}
for (int i=0; i<15;) {
   elemek[db].ertek=t[i];
   int j=i+1;
   while (j<15 && t[i]==t[j]) {
      j++;
   }
   elemek[db].szerepelt=j-i;
   db++;
   i=j;
}
```

```
struct Elemek{
   int szam;
   int darab=0;
};
...
const int n = 15;
int t[n];
...
int torol = t[0];
for (int i = 0;i < n;i++){
   if (t[i] > torol){
      torol = t[i];
   }
}
torol = torol + 1;
Elemek elem[n];
int db = 0;
for (int i = 0;i < n; i++){
   if (t[i] != torol){
      elem[db].szam = t[i];
      elem[db].darab = 1;

      for (int j = i+1; j < n; j++){
         if (t[j] == t[i]){
            t[j] = torol;
            elem[db].darab++;
         }
      }
      db++;
   }
}
```

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

using namespace std;

struct Utca{
   bool paratlan;
   int szeles;
   char kerites;
   int hazszam;
};

string intTOstrign(int);

int main()
{
   ifstream f("kerites.txt");
   Utca t[1000];
   int db=0;
   while (!f.eof()) {
      f >> t[db].paratlan >> t[db].szeles >> t[db].kerites;
      db++;
   }
   f.close();
   db--;
   for (int i=0; i<db; i++) {
      cout << t[i].paratlan << " " << t[i].szeles << " " << t[i].kerites << endl;
   }
   //2
   cout << endl;
   cout << "2. Feladat" << endl;
   cout << "    Telkek: " << db << endl;
```

```cpp
   cout << "3. Feladat" << endl;
   cout << "    Utolso telek oldala: ";
   int hazszamParos=2, hazszamParatlan=1;
   for (int i=0; i<db; i++) {
      if (t[i].paratlan) {
         t[i].hazszam=hazszamParatlan;
         hazszamParatlan+=2;
      } else {
         t[i].hazszam=hazszamParos;
         hazszamParos+=2;
      }
   }
   if (t[db-1].paratlan) cout << "paratlan";
      else cout <<"paros";
   cout << ", hazszama: " << t[db-1].hazszam << endl;
   //4
   cout << endl;
   cout << "4. Feladat" << endl;
   for (int i=0; i<db-1; i++) {
      if (isalpha(t[i].kerites) && t[i].paratlan) {
         int j;
         for (j=i+1; j<db; j++) {
               if (t[j].hazszam==t[i].hazszam+2) break;
         }
         if (t[i].kerites==t[j].kerites) {
               cout << "    A szomszedjanak ugyan olyan keritese van: " << t[i].hazszam << endl;
               break;
         }
      }
   }
```

```cpp
   cout << "5. Feladat" << endl;
   cout << "    Kerek egy hazszamot: ";
   int hazszam;
   cin >> hazszam;
   for (int i=0; i<db; i++) {
      if (t[i].hazszam==hazszam) {
         cout << "    Kerites: " << t[i].kerites << endl;
         int szomszedBal=-1, szomszedJobb=-1;
         for (int j=0; j<db; j++) {
               if (t[j].hazszam==t[i].hazszam-2) szomszedBal=j;
               if (t[j].hazszam==t[i].hazszam+2) szomszedJobb=j;
         }
         for (char j='A'; j<='Z'; j++) {
               if (szomszedBal==-1 && j!=t[i].kerites && j!=t[szomszedJobb].kerites) {
                  cout << "    Uj kerites: " << j << endl;
                  break;
               } else if (szomszedJobb==-1 && j!=t[i].kerites && j!=t[szomszedBal].kerites) {
                  cout << "    Uj kerites: " << j << endl;
                  break;
               } else if (j!=t[i].kerites && j!=t[szomszedJobb].kerites && j!=t[szomszedBal].kerites) {
                  cout << "    Uj kerites: " << j << endl;
                  break;
               }
         }
         break;
      }
   }
```

```cpp
   cout << "6. Feladat" << endl;
   string utca,keritesek;
   ofstream g("utca.txt");
   for (int i=0; i<db; i++) {
      if (t[i].paratlan) {
         keritesek+=intTOstrign(t[i].hazszam);
         for (int j=0; j<t[i].szeles; j++) {
               utca+=t[i].kerites;
               if (j>=intTOstrign(t[i].hazszam).length()) {
                  keritesek+=" ";
               }
         }
      }
   }
   g << utca << endl;
   g << keritesek << endl;
   return 0;
}

string intTOstrign(int sz) {
   string s1,s2;
   while (sz>0) {
      s1 += (char)(sz%10+48);
      sz /= 10;
   }
   for (int i=s1.length()-1; i>=0; i--) {
      s2+=s1[i];
   }
   return s2;
}
```

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

using namespace std;

struct Furdo{
   int azonosito, reszleg, kibe, ora, perc, mperc;
};

struct Furdo_vendeg{
   int azonosito=0, kibe=0, benttoltott_ido;
   int kezd_p, kezd_o, kezd_mp, veg_o, veg_p, veg_mp;
   int reszlegek[4]={};
};

int mpbe(int, int, int);

int main()
{
   ifstream f("furdoadat.txt");
   Furdo furdo[800];
   int furdo_db=0;
   while(!f.eof()){
      f >> furdo[furdo_db].azonosito >> furdo[furdo_db].reszleg;
      f >> furdo[furdo_db].kibe >> furdo[furdo_db].ora;
      f >> furdo[furdo_db].perc >> furdo[furdo_db].mperc;
      furdo_db++;
   }
   furdo_db--;
   f.close();
```

```cpp
   cout << endl << "2. feladat: " << endl;
   cout << "Az elso vendeg " << furdo[0].ora << ":" << furdo[0].perc << ":" << furdo[0].mperc << "-kor lepett ki az oltozobol." << endl;
   int vlm = furdo_db-2;
   while(furdo[vlm].reszleg!=0){
      vlm--;
   }
   cout << "Az elso vendeg " << furdo[vlm].ora << ":" << furdo[vlm].perc << ":" << furdo[vlm].mperc << "-kor lepett ki az oltozobol." << endl;

   cout << endl << "3. feladat: " << endl;
   int egyszer_db=0;
   Furdo_vendeg vendeg[100];
   int vendeg_db=0;
   for (int i=0; i<furdo_db; i++){
      bool kell = true;
      for(int j=0; j<vendeg_db; j++){
         if (furdo[i].azonosito == vendeg[j].azonosito){
               vendeg[j].kibe++;
               vendeg[j].veg_o = furdo[i].ora;
               vendeg[j].veg_p = furdo[i].perc;
               vendeg[j].veg_mp = furdo[i].mperc;
               kell = false;
               break;
         }
      }
      if (kell){
         vendeg[vendeg_db].azonosito = furdo[i].azonosito;
         vendeg[vendeg_db].kibe++;
         vendeg[vendeg_db].kezd_o = furdo[i].ora;
         vendeg[vendeg_db].kezd_p = furdo[i].perc;
         vendeg[vendeg_db].kezd_mp = furdo[i].mperc;
         vendeg_db++;
      }
   }
   for (int i=0; i<vendeg_db; i++){
      if (vendeg[i].kibe==4){
         egyszer_db++;
      }
   }
   cout << "A furdoben " << egyszer_db << " vendeg jart csak egy reszlegen." << endl;
```

```cpp
   cout << endl << "4. feladat: " << endl;
   for (int i=0; i<vendeg_db; i++){
      vendeg[i].benttoltott_ido = mpbe(vendeg[i].veg_o, vendeg[i].veg_p, vendeg[i].veg_mp) -  mpbe(vendeg[i].kezd_o, vendeg[i].kezd_p, vendeg[i].kezd_mp);
   }
   int maxi=0;
   for (int i=0; i<vendeg_db; i++){
      if (vendeg[maxi].benttoltott_ido<vendeg[i].benttoltott_ido)
         maxi=i;
   }
   cout << "A legtobb idot bent tolto vendeg: " << endl;
   int oraa =  vendeg[maxi].benttoltott_ido/3600;
   int peerc = (vendeg[maxi].benttoltott_ido - oraa*3600)/60;
   int masodpeerc = vendeg[maxi].benttoltott_ido - oraa*3600 - peerc*60;
   cout << vendeg[maxi].azonosito << ". vendeg " << oraa << ":" << peerc << ":" << masodpeerc << endl;

   cout << endl << "5. feladat: " << endl;
   int db6_9=0, db9_16=0, db16_20=0;
   for (int i=0; i<vendeg_db; i++){
      int kezdo = mpbe(vendeg[i].kezd_o, vendeg[i].kezd_p, vendeg[i].kezd_mp);
      if (mpbe(6, 0, 0) <= kezdo && kezdo < mpbe(9, 0, 0))
         db6_9++;
      if (mpbe(9, 0, 0) <= kezdo && kezdo < mpbe(16, 0, 0))
         db9_16++;
      if (mpbe(16, 0, 0) <= kezdo && kezdo < mpbe(20, 0, 0))
         db16_20++;

   }
   cout << "6 - 9 ora kozott: " << db6_9 << endl;
   cout << "9 - 16 ora kozott: " << db9_16 << endl;
   cout << "16 - 20 ora kozott: " << db16_20 << endl;
```

```cpp
   cout << endl << "6. feladat: " << endl;
   ofstream g("szauna.txt");
   for (int i=0; i<vendeg_db; i++){
      bool volt_szaunaban=false;
      int szaunaban=0;
      for (int j=0; j<furdo_db; j++){
         if (furdo[j].azonosito == vendeg[i].azonosito && furdo[j].reszleg == 2){
               volt_szaunaban=true;
               szaunaban += mpbe(furdo[j+1].ora, furdo[j+1].perc, furdo[j+1].mperc) - mpbe(furdo[j].ora, furdo[j].perc, furdo[j].mperc);
               j++;
         }
      }
      if (volt_szaunaban){
            oraa =  szaunaban/3600;
            peerc = (szaunaban - oraa*3600)/60;
            masodpeerc = szaunaban - oraa*3600 - peerc*60;
         g << vendeg[i].azonosito << " " << oraa << ":" << peerc << ":" << masodpeerc << endl;
      }
   }
   g.close();
```

```cpp
   cout << endl << "7. feladat: " << endl;
   for (int i=0; i<vendeg_db; i++){
      for (int j=0; j<furdo_db; j++){
         if(furdo[j].azonosito==vendeg[i].azonosito){
               vendeg[i].reszlegek[furdo[j].reszleg-1]++;
         }
      }
   }
   int uszoda=0, szauna=0, gyogyviz=0, strand=0;
   for (int i=0; i<vendeg_db; i++){
      for (int j=0; j<4; j++){
         if (vendeg[i].reszlegek[j]!=0){
            switch(j){
            case 0: uszoda++;
               break;
            case 1: szauna++;
               break;
            case 2: gyogyviz++;
               break;
            case 3: strand++;
               break;

            }
         }
      }
   }
   cout << "Uszoda: " << uszoda << endl;
   cout << "Szaunak: " << szauna << endl;
   cout << "Gyogyvizes medencek: " << gyogyviz << endl;
   cout << "Strand: " << strand << endl;
   ///0 - be, elhagyja az uyzodat; 1 - ki, bemegy az uszodaba;
   return 0;
}

int mpbe(int o, int p, int mp){
   return 3600*o+60*p+mp;
}
```