

Задача E1. Числа

По време на дългите учебни часове нашият приятел Умко от училището в село Каръшко, което в наша чест е наречено “Програмистите на България”, се забавлявал като измислял различни игри с числа. Случайно забелязал, че от едно петцифрено число, чрез разместване на цифрите му, могат да се получат много различни числа и дори ги преброил. Те били 120. Все пак написването им било досадна работа, дори когато ти е скучно в час. Умко решил, че не го интересуват всички тези числа, а само най-малкото и най-голямото от тях. Той започнал да разглежда различни петцифрени числа и за всяко от тях да намира разликата между най-голямото и най-малкото число, получени от неговите цифри. Скоро и тази работа му омръзнала, но все пак искал на всяка цена да знае тази разлика за всяко едно петцифрено число.

Сега вече само вие можете да му помогнете като напишете програма **NUMBERS.EXE**, която въвежда от клавиатурата на компютъра цяло петцифрено число N и извежда на екрана разликата между най-малкото и най-голямото измежду числата, получени като се разместят цифрите на N.

Примерен вход:

56342

Примерен изход:

41976

Обяснение:

Най-голямото число, което се получава от цифрите 2, 3, 4, 5 и 6 на числото 56342, е 65432, най-малкото е 23456, а разликата на тези две числа е точно 41976.

Примерен вход:

10002

Примерен изход:

19989

Обяснение:

Най-голямото число, което се получава от цифрите 1, 0, 0, 0 и 2 на числото 10002, е 20001, най-малкото е 12, а разликата на тези две числа е точно 19989.

Решение:

I начин: (Когато учениците не са учили масиви и оператори за цикъл.)

1. Необходими величини:

Три цели петцифрени променливи, които ще са от тип **long**, защото в тип **int** не се включват всички цели петцифрени числа, а само тези до 32767. В едната от тях ще се въведе числото N, а в другите две съответно минималното и максималното число, които се получават от цифрите на N.

```
long N, maxN, minN;
```

Освен тези три променливи ще ни трябва по една за всяка една от цифрите на числото N, които вече е добре да са от тип **int**.

```
int e, d, s, h, dh;
```

2. Въвежда се числото

```
cin >> N;
```

3. Пресмятат се цифрите му:

```
e=N%10;
```

```
d=N/10%10;
```

```
s=N/100%10;
```

```
h=N/1000%10;
```

```
dh=N/10000;
```

4. Подреждат се петте цифри по големина, като най-голяма става цифрата на десетохилядните, а най-малка цифрата на единиците:

```
if (e>d) { int z=e; e=d; d=z; }
if (d>s) { z=s; s=d; d=z; }
if (s>h) { z=s; s=h; h=z; }
if (h>dh) { z=h; h=dh; dh=z; }
if (e>d) { z=e; e=d; d=z; }
if (d>s) { z=s; s=d; d=z; }
if (s>h) { z=s; s=h; h=z; }
if (e>d) { z=e; e=d; d=z; }
if (d>s) { z=s; s=d; d=z; }
if (e>d) { z=e; e=d; d=z; }
```

5. Получава се по-малкото от числата:

```
minN=e*10000+d*1000+s*100+h*10+dh;
```

6. Получава се по-голямото от числата:

```
maxN=dh*10000+h*1000+s*100+d*10+e;
```

7. Извежда се разликата между максималното и минималното число:

```
cout<<maxN-minN<<endl;
```

II начин: (Когато учениците са учили масиви и оператори за цикъл.)

1. Необходими величини:

Три цели петцифрени променливи, които ще са от тип **long**, защото в тип **int** не се включват всички цели петцифрени числа, а само тези до 32767. В едната от тях ще се въведе числото N, а в другите две съответно минималното и максималното число, които се получават от цифрите на N.

```
long N, maxN, minN;
```

Този път цифрите на числото записваме в масив от пет елемента:

```
int a[5], i, j;
```

2. Въвежда се числото

```
cin>>N;
```

3. Пресмятат се цифрите му:

```
for ( i=0; i<5; i++) { a[i]=N%10; N/=10; }
```

4. Подреждат се петте цифри по големина, като най-голяма става цифрата на десетохилядните, а най-малка цифрата на единиците:

```
for ( i=0; i<4; i++)
    for ( j=0; j<4; j++)
        if (a[j]>a[j+1]) { int z=a[j]; a[j]=a[j+1]; a[j+1]=z; }
```

5. Получават се двете числа:

```

maxN=minN=0;
for ( i=0; i<5; i++)
{
    minN=minN*10+a[i];
    maxN=maxN*10+a[4-i];
}

```

6. Извежда се разликата между максималното и минималното число:

```

cout<<maxN-minN<<endl;

```

Ето окончателния вид на програмата от втория начин:

```

#include<iostream.h>
void main ()
{
    long N, maxN, minN;
    int a[5], i, j;
    cin >> N;
    for ( i=0; i<5; i++) {a[i]=N%10; N/=10; }
    for ( i=0; i<4; i++)
        for (j=0; j<4; j++)
            if (a[j]>a[j+1]) { int z=a[j]; a[j]=a[j+1]; a[j+1]=z; }
    maxN = minN = 0;
    for ( i=0; i<5; i++)
    {
        minN=minN*10+a[i];
        maxN=maxN*10+a[4-i];
    }
    cout<<maxN-minN<<endl;
}

```

Задача E2. Бонбонки

Умко (нашият познайник от училище “Програмистите на България”) имал странно хоби. Той имал четири кристални купички, в които грижливо събирал шоколадови бонбонки. Странното било това, че той не обичал да си похапва от тях, а само да си ги гледа разпределени по равен брой в своите купички. Във всяка от тях той държал по 10 бонбонки. Това, че нашето умно момче не обичало шоколадовите бонбонки съвсем не означавало, че и неговата сестра не обичала да си похапва сладко, сладко от тях. Тя само изчаквала Умко да отиде на училище и с огромно удоволствие си похапвала. Така всеки ден след като се връщал от училище, той виждал, че любимите му кристални купички вече не са пълни с по 10 бонбонки във всяка и тичал до магазина, за да набави необходимите бонбонки. След известно време Умко сменил тактиката. Вместо всеки ден да ходи до магазина и да си купува бонбонки, просто ги премествал от едната купичка в другата, за да станат пак по равен брой в четирите купички. В случаите, когато това не било възможно, Умко все пак тичал до магазина.

Тъй като му омръзнало всеки път да пресмята колко бонбони трябва да размести и да купи, нашият приятел ви моли да напишете програма **BONBONKI.EXE**, която прочита от клавиатурата последователно броя на бонбонките във всяка една от четирите купички и ако е възможно те да се разпределят по равно, извежда на екрана от коя купичка колко бонбонки се изваждат или добавят. Ако бонбонките се изваждат от купичката, пред броя им се поставя знака “-“, а ако те се добавят – знака “+”. Ако броя на бонбонките в съответната купичка остава непроменен, се извежда числото 0 без знак. Ако не е възможно бонбонките да се разпределят по равно в четирите купички, програмата извежда на екрана само броя на бонбонките, които Умко трябва да купи от магазина, за да поправи разпределението им.