INT to TEXT

Michał Jakub Krupa

Zadanie 1.

1. Określić w jaki sposób przechowujemy daty
   Datę przechowujemy w zmiennej int, jako „sklejony rok z miesiącem i dniem” – to znaczy rok*10 000 + miesiąc* 100 + dzień.

2. Zaproponować algorytm
   Wybrałem sortowanie pozycyjne. Jego przewagą jest fakt, że nie wykonuje żadnych operacji porównania na danych wejściowych. Optymalny dla dużej ilości długich liczb podobnych do siebie – w tym sensie, że większość z nich ma takie same cyfry na początkowych pozycjach.

3. Określić złożoność
   Złożoność : O(d(n+k)) = O(8(40mln+10)) = 320000080

4. Opis algorytmu:

Tablica o długości 40 milionów z liczbami z zakresu (w zależności od dat urodzenia) 19000101 do 20210922 (dnia dzisiejszego) zostaje pobrana z naszej bazy danych. Ja wybrałem sortowanie rosnące. Pierwszym krokiem będzie posortowanie tablicy przez sortowanie przez zliczanie i kluczem w tym sortowaniu będzie ostatnia cyfra każdej tej liczby.

To sortowanie zlicza ilość wystąpień danej cyfry do 9 elementowej tablicy, z uwagi na to że nasz przedział to [0-9]. Zaczynając od drugiego elementu, kolejno dodajemy wartość z poprzedniego. Teraz przesuwamy tablicę o jeden element w prawo (w pierwszej komórce pojawi się zero a ostatni zostaje nadpisany i nie przechodzi dalej). Następnie tworzona jest docelowa tablica o długości 40 milionów. Teraz bierzemy każdą wartość elementu naszej startowej tablicy i zapisujemy w wyjściowej tablicy pod indexem, który jest wartością naszej 9 elementowej pomocniczej tablicy, spod indexu równego przenoszonej cyfrze. Następnie inkrementujemy wartość w tablicy pomocniczej, która służyła nam za indeks w docelowej tablicy. Procedurę powtarzamy, aż przeniesiemy wszystkie cyfry. Pierwsze sortowanie przez zliczanie zostało zakończone.

Następnie klucz przesuwa się po kolejnych cyfrach liczb z tablicy (musi posortować 8 pozycji cyfr) wywołując kolejny raz sortowanie przez zliczanie.

Zadanie 2.

1. Kod programu:

```
#include <iostream>
using namespace std;

string nazwaTrzech(long long x) {
   string jednosci[10] = { "zero","jeden ","dwa ","trzy ","cztery ","piec ","szesc ","siedem ","osiem ","dziewiec " };
   string nastki[11] = {"","dziesiec ","jedenascie ","dwanascie ","trzynascie ","czternascie ","pietnascie ","szesnascie
","siedemnascie ", "osiemnascie ","dziewietnascie " };
   string dziesiatki[10] = { "","dziesiec ","dwadziescia ","trzydziesci ","czterdziesci ","piecdziesiat ","szescdziesiat
","siedemndziesiat ", "osiemdziesiat ","dziewiecdziesiat " };
   string setki[10] = { "","sto ","dwiescie ","trzysta ","czterysta ","piecset ","szescset ","siedemset ","osiemset ","dziewiecset
" };  //////////////////////
   int liczba = x, koncowka;
   string slownie = "";

   for(int i = 0; liczba > 0; i++)
   {
      koncowka = liczba % 10;
      liczba = liczba / 10;

      if (i == 0 && (liczba % 10 != 1))
      {
         slownie = jednosci[koncowka] + slownie;
      }

      if (i == 0 && (liczba % 10 == 1))
      {
         slownie = nastki[koncowka + 1] + slownie;
         liczba = liczba / 10;
      }

      if (i == 1)
      {
         slownie = dziesiatki[koncowka] + slownie;
      }

      if (i == 2)
      {
         slownie = setki[koncowka] + slownie;
      }
   }

   return slownie;
}

void wypiszSlownie(long long x) {
   if (x >= 0 && x <= 999999999999) {
      int dlugosc = 0;
      long long liczba[4];
      long long pomoc = x;
      long long pomoc2 = x;
      string cala = "";
      for (int i = 1; pomoc >= 1; i++)
      {
         pomoc = pomoc / 10;
         dlugosc = i;
      }
      for (int i = 0; pomoc2 >= 1; i++)
      {
         liczba[i] = pomoc2 % 1000;
```

```
            pomoc2 = pomoc2 / 1000;
        }
        for (int i = 3; i >= 0; i--)
        {

            if (i == 0)
            {
                cala = cala + nazwaTrzech(liczba[0]);

            }
            else if (i == 1 && dlugosc > 3)
            {
                cala = cala + nazwaTrzech(liczba[1]) + " tys. ";

            }
            else if (i == 2 && dlugosc > 6)
            {
                cala = cala + nazwaTrzech(liczba[2]) + " mln. ";

            }
            else if (i == 3 && dlugosc > 9)
            {
                cala = cala + nazwaTrzech(liczba[3]) + " mld. ";

            }
        }
        cout << cala;
    }
    else if (x < 0) {
        cout << "Podano za mala liczbe!";
    }else if(x>999999999999)
    {
        cout << "Podano za duza liczbe!";
    }
}
int main()
{
    wypiszSlownie(123333444555);
}
```

2. Opis algorytmu

Funkcja wypiszSlownie przyjmuje 1 argument typu long long (nie int ponieważ funkcja ma wypisywać także miliardy (12 cyfr), a int jest ograniczony do 9 cyfr).

```
void wypiszSlownie(long long x)
```

Sprawdzamy czy argument podany w funkcji jest większy od zera oraz czy jest maksymalnie 12 cyfrowy.

```
if (x >= 0 && x <= 999999999999) { ... }
else if (x < 0) {
    cout << "Podano za mala liczbe!";
}else if(x>999999999999)
{
    cout << "Podano za duza liczbe!";
}
```

W pętli for sprawdzamy długość naszej liczby. Robimy to dzieląc liczbe przez dziesiec. Wykorzystujemy tutaj fakt, że int nie jest liczbą zmiennoprzecinkową.

```
long long pomoc = x;
for (int i = 1; pomoc >= 1; i++)
{
    pomoc = pomoc / 10;
    dlugosc = i;
}
```

Następnie w pętli rozdzielamy naszą liczbę na trzy trójliczbowe segmenty, zapisując je kolejno w tablicy.

```
long long pomoc2 = x;
for (int i = 0; pomoc2 >= 1; i++)
{
    liczba[i] = pomoc2 % 1000;
    pomoc2 = pomoc2 / 1000;
}
```

W kolejnej pętli w każdym segmencie który nie jest pusty, używamy funkcji nazwaTrzech, która zamienia liczby na słowa oraz w zależności od numeru segmentu dopisuje odpowiednie skróty (mld., mln., tys.).

```
for (int i = 3; i >= 0; i--)
{

    if (i == 0)
    {
        cala = cala + nazwaTrzech(liczba[0]);

    }
    else if (i == 1 && dlugosc > 3)
    {
        cala = cala + nazwaTrzech(liczba[1]) + " tys. ";

    }
    else if (i == 2 && dlugosc > 6)
    {
        cala = cala + nazwaTrzech(liczba[2]) + " mln. ";

    }
    else if (i == 3 && dlugosc > 9)
    {
        cala = cala + nazwaTrzech(liczba[3]) + " mld. ";

    }
}
cout << cala;
```

Robimy to oddzielając liczbę jedności od setek i dziesiątek.

```
koncowka = liczba % 10;
liczba = liczba / 10;
```

W zależności od warunku do naszego zwracanego stringa dopisywane są słowa z wcześniej przygotowanych tablic.

```
for(int i = 0; liczba > 0; i++)
{
    koncowka = liczba % 10;
    liczba = liczba / 10;

    if (i == 0 && (liczba % 10 != 1))
    {
        slownie = jednosci[koncowka] + slownie;
    }

    if (i == 0 && (liczba % 10 == 1))
    {
        slownie = nastki[koncowka + 1] + slownie;
        liczba = liczba / 10;
    }

    if (i == 1)
    {
        slownie = dziesiatki[koncowka] + slownie;
    }

    if (i == 2)
    {
        slownie = setki[koncowka] + slownie;
    }
}
```

```
string jednosci[10] = { "zero","jeden ","dwa ","trzy ","cztery ","piec ","szesc ","siedem ","osiem ","dziewiec " };

string nastki[11] = {"","dziesiec ","jedenascie ","dwanascie ","trzynascie ","czternascie ","pietnascie ",
   "szesnascie ","siedemnascie ", "osiemnascie ","dziewietnascie " };

string dziesiatki[10] = { "","dziesiec ","dwadziescia ","trzydziesci ","czterdziesci ","piecdziesiat ",
   "szescdziesiat ","siedemndziesiat ", "osiemdziesiat ","dziewiecdziesiat " };

string setki[10] = { "","sto ","dwiescie ","trzysta ","czterysta ","piecset ","szescset ","siedemset ","osiemset ","dziewiecset " };
```

Wynik tej funkcji jest doklejany do naszego ostatecznego stringa, który jest wypisywany.