

ZAGADNIENIA

- Zapisywanie danych do pliku
- Otwieranie pliku do odczytu
- Odczytywanie danych z pliku
- Tryby otwarcia pliku

Narzędzia do pracy z plikami znajdują się w bibliotece **fstream** (termin pochodzi od angielskich słów **file stream** – ‘strumień plikowy’), który dołączamy na początku programu za pomocą dyrektywy **#include <fstream>**. Aby móc odczytywać lub zapisywać dane do pliku, w pierwszej kolejności należy zdefiniować zmienną, za której pomocą będziemy mogli wykonywać te operacje. W tym celu wykorzystamy klasę **fstream**, która jest umieszczona w przestrzeni nazw **std**.

Operacje związane z przetwarzaniem pliku składają się z etapów, którymi są:

- zdefiniowanie strumienia, czyli stworzenie obiektu klasy **fstream**;
- otwarcie pliku;
- wykonanie operacji na pliku;
- zamknięcie pliku.

Zapisywanie danych do pliku tekstowego

Jak już wcześniej wspomniano, na początku programu należy dodać bibliotekę **fstream**, a następnie – utworzyć zmienną plikową. Zmienna umożliwi nam pracę na danym pliku. Jeśli chcemy zapisać dane do pliku, to najpierw należy go otworzyć, jeśli on istnieje, lub utworzyć, jeżeli na dysku nie ma pliku o takiej nazwie.

PRZYKŁAD 36.1

Utwórzmy program, który poprosi o podanie imienia i znaku zodiaku użytkownika. Te dane zostaną zapisane w pliku `dane.txt`, znajdującym się w folderze `dane` na dysku C:.

```
#include <iostream>
#include <fstream>
using namespace std;
string imie, zodiak;
int main()
{
```

```

cout<<"Podaj imie: ";
cin>>imie;
cout<<"Podaj znak zodiaku: ";
cin>>zodiak;
//zmienna plikowa
fstream plik;
//otwarcie pliku do zapisu
plik.open("c:/dane/dane.txt",ios::out);
//zapisanie imienia do pliku
plik<<imie<<endl;
//zapisanie znaku zodiaku do pliku
plik<<zodiak<<endl;
//zamkniecie pliku
plik.close();
return 0;
}

```

Po wprowadzeniu wartości dla dwóch zmiennych utworzono zmienną plikową o nazwie **plik**. Na rzecz tej zmiennej została wywołana metoda `open`, której pierwszym argumentem jest ścieżka dostępu do pliku. Ścieżka do pliku może być **względna** (określająca położenie pliku względem miejsca, w którym znajduje się program) lub **bezwzględna** (odnosząca się do katalogu głównego na dysku).

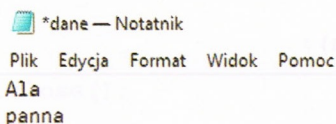
Drugi argument `ios::out` informuje kompilator, w jakim trybie ma zostać otwarty plik. Parametr `out` oznacza, że plik jest otwarty do zapisu.

Lista niektórych dostępnych trybów:

- `ios::app` – tryb umożliwiający dopisywanie danych do istniejących danych w pliku;
- `ios::in` – tryb umożliwiający odczytywanie danych z pliku;
- `ios::out` – tryb umożliwiający zapisywanie danych do pliku.

Po otwarciu pliku następuje zapisywanie do niego określonych wartości. Instrukcja jest bardzo prosta. Jeśli chcemy zapisać wartość zmiennej do pliku, wystarczy słowo `cin` zastąpić nazwą zmiennej plikowej. Po wykonaniu wszystkich operacji na pliku należy go zamknąć za pomocą metody `close`.

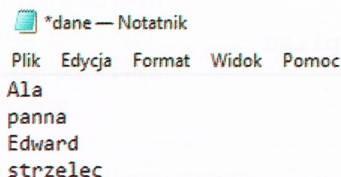
Po uruchomieniu programu i wprowadzeniu danych utworzy się plik o nazwie **dane.txt**. Będą w nim zawarte dane wprowadzone przez użytkownika.



Niestety, po ponownym uruchomieniu programu i podaniu nowych danych okaże się, że poprzednie dane zostały nadpisane. Aby umożliwić dopisanie kolejnych danych do plików, wystarczy użyć dodatkowej instrukcji do metody `open` – modyfikatora `app`, który oznacza „dołącz” dane do pliku istniejącego:

```
plik.open("c:/dane/dane.txt",ios::out| ios::app);
```

Ta niewielka korekta programu spowoduje, że jeśli ponownie uruchomimy program i wprowadzimy inne dane, to zostaną one dopisane na koniec do pliku. Przykład pokazano na rys. 36.2.



```
*dane — Notatnik
Plik  Edycja  Format  Widok  Pomoc
Ala
panna
Edward
strzelec
```

Rys. 36.2. Dopisanie danych do pliku `dane.txt`

Odczytywanie zawartości pliku

Tym razem użyjemy metody `open` z modyfikatorem `in`, co oznacza, że plik otwieramy do odczytu danych.

Otwarcie pliku może zakończyć się zarówno powodzeniem, jak i fiaskiem. Zanim zaczniemy pracować na danych z pliku, warto sprawdzić, czy udało się otworzyć plik. Do tego celu należy użyć metody `good`, która należy do klasy `fstream`. Funkcja ta zwraca wartość `TRUE`, gdy plik istnieje, lub wartość `FALSE`, gdy plik nie istnieje.

! UWAGA

Wiersze w pliku są numerowane od 1, a nie od zera – jak w tablicach.

PRZYKŁAD 36.2

```
#include <iostream>
#include<fstream>
using namespace std;
int main()
{
    fstream plik;
    plik.open("c:/dane.txt",ios::in);
    if(plik.good()==true)
    {
        cout<<"Udalo sie otworzyc plik";
    }
}
```

Na powyższym przykładzie widać, że jeśli plik istnieje, otrzymamy komunikat, iż udało się go otworzyć, i będziemy mogli przejść do etapu odczytania jego zawartości. Do odczytywania zawartości pliku stosujemy funkcję **getline**. Jej zadaniem jest wczytywanie kolejnych wierszy pliku oraz zwrócenie wartości logicznej, która informuje, czy można kontynuować odczyt. Jeśli funkcja zwróci wartość TRUE, to znaczy, że plik nie został jeszcze w całości odczytany i można dalej odczytywać dane z pliku. Jeśli natomiast funkcja zwróci FALSE, to znaczy, że został osiągnięty koniec pliku albo wystąpił błąd. Oczywiście odczyt za pomocą funkcji **getline** musi odbywać się w pętli.

PRZYKŁAD 36.3

```
#include <iostream>
#include<fstream>
using namespace std;
int main()
{
    fstream plik;
    plik.open("c:/dane/dane.txt",ios::in);
    if( plik.good()==false)
    {
        cout<<"Nie udalo sie otworzyc pliku";
    }
    else
    {
        string wiersz;
        while(getline(plik,wiersz))
        {
            cout<<wiersz<<endl;
        }
    }
    plik.close();
    return 0;
}
```