Temat: Operacje na plikach.

Operacje tego typu umożliwia nam biblioteka **fstream**, dzięki której uzyskujemy funkcje pozwalające nam zarówno zapisywać pliki jak i je odczytywać.

Zanim zaczniemy odczytywać, bądź zapisywać dane z/do pliku, musimy posiadać zmienną, dzięki której będziemy mogli wykonywać operacje na wybranym pliku. W tym celu utworzona została klasa **fstream**. Klasa ta jest umieszczona w przestrzeni nazw **std::**. Klasa ta udostępnia nam cały interfejs, dzięki któremu będziemy mogli obsłużyć dowolny plik znajdujący się na dysku lub innym nośniku danych.

```
std::fstream plik;
```

Oczywiście jeżeli przestrzeń nazw została wcześniej zadeklarowana poprzez:

```
using namespace std;
```

To obsługę pliku (zmienną) inicjujemy wpisując:

```
fstream plik;
```

1. Otwieranie pliku

Zmienna, utworzona aktualnie nie wskazuje na żaden plik. Aby przypisać konkretny plik do zmiennej wywołujemy funkcję **open()**, której definicja wygląda następująco:

```
void open( const char * nazwa_pliku, ios_base::openmode tryb_otwarcia_pliku );
```

Pierwszy parametr funkcji (**nazwa_pliku**) określa ścieżkę dostępu i nazwę pliku do jakiego chcemy uzyskać dostęp. Drugi parametr funkcji, czyli **tryb_otwarcia_pliku** służy do poinformowania kompilatora w jakim trybie dany plik chcemy otworzyć. Lista dostępnych trybów wraz z opisami w poniższej tabeli.

Tryb	Opis trybu
ios::app	(**app**end - dopisywanie danych do pliku) Ustawia wewnętrzny wskaźnik zapisu pliku na jego koniec. Plik otwarty w trybie tylko do zapisu. Dane mogą być zapisywane tylko i wyłącznie na końcu pliku.
ios::ate	(**at** end) Ustawia wewnętrzny wskaźnik pliku na jego koniec w chwili otwarcia pliku.
ios::binary	(**binary**) Informacja dla kompilatora, aby dane były traktowane jako strumień danych binarnych, a nie jako strumień danych tekstowych.
ios::in	(**in**put - wejście/odczyt) Zezwolenie na odczytywanie danych z pliku.
ios::out	(**out**put - wyjście/zapis) Zezwolenie na zapisywanie danych do pliku.
ios::trunc	(**trunc**ate) Zawartość pliku jest tracona, plik jest obcinany do 0 bajtów podczas otwierania.

Wszystkie wymienione tryby możemy łączyć ze sobą - oznacza to, że jeśli chcemy otrzymać plik do odczytu i zapisu wystarczy oddzielić je pojedynczym operatorem |.

```
fstream plik;
plik.open( "nazwa_pliku.txt", std::ios::in | std::ios::out );
```

a) Czy udało otworzyć się plik?

Warto zawsze sprawdzić czy plik został otwarty prawidłowo.

Po wykonaniu operacji otwarcia pliku, wewnątrz klasy ustawiane są odpowiednie flagi, które informują o tym, czy otrzymaliśmy dostęp do pliku czy też nie. Funkcje, jakie umożliwiają nam sprawdzenie tego stanu to **good()** oraz **is_open()**. Definicja tych funkcji wygląda następująco:

```
bool good();
bool is_open();
```

Obie funkcje zwrócą wartość **true**, jeśli uzyskano dostęp do pliku, w przeciwnym wypadku otrzymamy wartość **false**.

```
fstream plik;
plik.open( "nazwa_pliku.txt", std::ios::in | std::ios::out );
if( plik.good() == true )
{
  cout << "Uzyskano dostep do pliku!" << std::endl;
   //tu operacje na pliku
}
else cout << "Dostep do pliku zostal zabroniony!" << std::endl;
```

2. Zamykanie pliku

Każdy plik należy zamykać po zakończeniu pracy z nim. Jeśli plik ma być używany tylko przez jednego użytkownika szkodliwość jest stosunkowo mała - klasa **fstream** sama zamknie plik przed usunięciem zmiennej z pamięci. Jeśli natomiast zapomnisz zamknąć plik, którego dane mają być współdzielone przez kilku użytkowników, automatycznie uniemożliwisz im dostęp do tego zasobu. Funkcja odpowiedzialna na zamykanie pliku nosi nazwę **close()**. Deklaracja wygląda następująco:

```
void close( void );
```

```
#include <fstream>
using namespace std;
int main()
{
  fstream plik;
   plik.open( "nazwa_pliku.txt", ios::in | ios::out );
   if( plik.good() == true )
   {
      //tu operacje na pliku (zapis/odczyt)
      plik.close();
   }
   return( 0 );
}
```

3. Odczytywanie danych z pliku

Jeśli uzyskamy już dostęp do pliku w trybie do odczytu, możemy rozpocząć odczytywanie danych z pliku. Język C++ oferuje więcej niż jedną metodę odczytu danych z pliku.

a. Pobieranie danych za pomocą strumienia

Pierwszą, a zarazem bardzo wygodną metodą odczytywania danych z pliku jest strumień. Ponieważ zapis jest analogiczny do strumienia **std::cin>>**, przedstawiam tylko formę zapisu.

```
nazwa_zmiennej_plikowej >> zmienna_do_ktorej_dane_maja_zostac_zapisane;
```

Co należy wiedzieć o strumieniu:

- Dane odczytywane za pomocą strumienia są zawsze traktowane jako tekst, niezależnie czy podczas otwierania użyliśmy trybu ios::binary czy nie.
- Strumień działa analogicznie do std::cin>>, co w konsekwencji oznacza, że za pomocą tej funkcji nie odczytamy żadnej informacji o białych znakach (tj. enter, tabulacja, spacja itp).

b. Odczytywanie danych do końca pliku

Funkcja służy do sprawdzania, czy wskaźnik pliku znajduje się na końcu pliku. Definicja funkcji:

```
bool eof();
```

Funkcja zwróci wartość true wtedy, gdy nie będzie już w pliku więcej danych do odczytu. Dzięki tej funkcji możemy w bardzo łatwy sposób odczytać zawartość całego pliku.

Poniższy przykład wyświetli zawartość całego pliku na ekran konsoli.

Uwaga: plik dane.txt musi być umieszczony w tym samym folderze w którym wskazujemy umieszczenie pliku źródłowego naszego programu. W innej sytuacji należy wskazać pełną ścieżkę dostępu do pliku. Plik ten musicie utworzyć, można użyć notatnika zapisując w nim dowolny tekst.

#include <conio.h>

```
#include <iostream>
#include <fstream>
using namespace std;
int main()
{
    fstream plik;
    plik.open( "dane.txt", ios::in );
    if( plik.good() )
    {
        string znak;
        cout << "Zawartosc pliku:" << endl;
        while( !plik.eof() )
        {
            plik >> znak;
            cout << znak << endl;
        }
        plik.close();
    } else cout << "Error! Nie udalo otworzyc sie pliku!" << endl;

    getch();
    return( 0 );
}
```

Ćwiczenie:

1. Proszę wykonać powyższy program. Proszę sprawdzić działanie programu zmieniając typ zmiennej znak na int, char.
2. Proszę w pliku tekstowym umieścić w kolejnych 5 wierszach liczby całkowite mniejsze od 20, plik zapisać pod nazwą: dane-l.txt.
   Napisz program który odczytuje kolejne liczby z pliku dane-l.txt i oblicza ich iloczyn. Iloczyn wyświetlany jest po zamknięciu pliku dane-l.txt. Nazwa programu: Iloczyn_L
3. Proszę w pliku tekstowym umieścić w kolejnych 10 wierszach pojedyncze słowa, plik zapisać pod nazwą slowa.txt.
   Napisz program który odczytuje kolejne słowa z pliku slowa.txt i umieszcza je w dziesięcioelementowej tablicy. Elementy tablicy wyświetlane są na ekranie po zamknięci pliku slowa.txt. Nazwa programu: Tablica_S

Pliki ćwiczeń z całymi katalogami zawierającymi kody, źródłowe pliki tekstowe i dołączonymi bibliotekami proszę spakować (rar lub zip) i przesłać do 15 min po zakończonej lekcji na adres: marek@zstio-elektronika.pl
Pliki przysłane po terminie, lub ich brak, ocenione zostaną na ndst bez możliwości poprawy.