1 Strumienie

W języku C++ pliki obsługiwane są za pomocą *strumieni*. Strumień pozwala na sekwencyjny dostęp do pliku. Elementy, które jako pierwsze zostają zapisane do strumienia jako pierwsze będą z niego odczytane. Strumienie wykorzystywane są także do obsługi standardowego wejścia/wyjścia (std::cin, std::cout, std::cerr, std::clog). Zapisanie danych do strumienia odbywa się za pomocą operatora <<, natomiast odczytanie danych ze strumienia za pomocą operatora >>. Operatory te zwracają strumień na rzecz którego zostały wywołane dlatego możliwe jest np jednoczesne wysłanie kilku elementów do strumienia. Przykład:

```
int i = 5;
std::cout << "Tekst" << i << "Inny tekst" << std:: end;
```

2 Pliki

Strumienie do obsługi plików zdefiniowane są w pliku nagłówkowym fstream. W pliku tym zdefiniowane są następujące strumienie:

- std::ifstream – strumień do odczytu pliku
- std::ofstream – strumień do zapisu pliku
- fstream – strumień do odczytu i zapisu pliku

2.1 Zapis do pliku tekstowego

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

int main(){

std::ofstream file;
file.open("plik.txt");
file << "Hello World";
file.close();

return 0;
}
```

2.2 Odczyt pliku tekstowego

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

int main(){

std::string linia;
std::ifstream plik;
plik.open("plik.txt");

if (plik.is_open()){
        while(getline(plik, linia))
                std::cout << linia << std::endl;
        plik.close();

} else std::cout << "Nie udalo sie otworzyc pliku" << std::endl;

return 0;
}
```

2.3 Zapis pliku binarnego

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

int main(){

int i = 3;
float f = 4.6f;
long l = 30000001;

std::ofstream plik;
plik.open("plik.abc", std::ios::binary);
plik.write(reinterpret_cast<char*>(&i), sizeof(i));
plik.write(reinterpret_cast<char*>(&f), sizeof(f));
plik.write(reinterpret_cast<char*>(&l), sizeof(l));
plik.close();

return 0;
}
```

2.4 Odczyt pliku binarnego

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

int main(){

int i;
float f;
long l;

std::ifstream plik;
```

```
plik.open("plik.abc", std::ios::binary);
plik.read(reinterpret_cast<char*>(&i), sizeof(i));
plik.read(reinterpret_cast<char*>(&f), sizeof(f));
plik.read(reinterpret_cast<char*>(&l), sizeof(l));
plik.close();
std::cout << i << "," << f << "," << l << std::endl;

return 0;
}
```

Pierwsze parametry metod read i write to wskaźniki na adres w pamięci gdzie znajdują się dane do zapisania. Dlatego aby przesłać podstawowe typy muszą one wcześniej zostać rzutowane na typ char*.

2.5 Metody strumieni

- open() – otwiera plik do zapisu lub odczytu. Jako drugi parametr można podać opcjonalne parametry:
  - std::ios::binary – otwieramy plik binarny, jeśli brak parametru to plik tekstowy
  - std::ios::in – otwieramy plik do odczytu, jeśli otwieramy strumień ifstream to parametr ten można pominąć
  - std::ios::out – otwieramy plik do zapisu, jeśli otwieramy strumień ofstream to parametr ten można pominąć
  - std::ios::ate – otwieramy plik i przesuwamy się na koniec pliku w celu dopisania zawartości (można zmienić pozycję wskaźnika)
  - std::ios::app – otwieramy plik tylko i wyłącznie do dopisywania zawartości (nie można zmienić pozycji wskaźnika w celu nadpisania)
  - std::ios::trunc – otwieramy plik wymazując jego poprzednią zawartość
- close() – zamyka plik
- is_open() – sprawdza czy plik został prawidłowo otworzony
- read() – odczytuje dane z pliku
- write() – zapisuje dane do pliku
- getline() – pobiera całą linię z pliku
- tellg(), tellp() – zwraca aktualną pozycję w pliku odpowiednio do pobierania i zapisywania

- seekg(), seekp() – ustawia aktualną pozycję w pliku odpowiednio do pobierania i zapisywania
- eof() – sprawdza czy osiągnięto koniec pliku
- fail() – sprawdza czy operacja zakończyła się niepowodzeniem
- bad() – sprawdza czy operacja zakończyła się problemem uniemożliwiającym dalsze operacje na strumieniu
- good() – sprawdza czy strumień jest prawidłowy

3 Strumienie łańcuchów znaków

W programach pisanych w C++ przydatny może być także strumienie operujące na łańcuchach znaków zdefiniowane w <sstream>:

- istringstream – wejściowy strumień łańcucha znaków
- ostringstream – wyjściowy strumień łańcucha znaków
- stringstream – wejściowo/wyjściowy strumień łańcucha znaków

Strumień ostringstream może być przydatny np. do zamiany określonych typów danych na postać łańcucha znaków:

```
std::ostringstream oss;
oss << "Hello" << i << ";" << l;
std::cout << oss.str() << std::endl
```

Strumień istringstream może być przydatny np. do pobrania określonych wartości z łańcucha znaków:

```
std::string str = "1 2 3";
std::istringstream iss(str);
int a1, a2, a3;
iss >> a1 >> a2 >> a3;
std::cout << a1 << "," << a2 << "," << a3 << std::endl;
```

4 Zadania do wykonania

1. Zapisać do pliku tekstowego linie pobrane od użytkownika do czasu aż wpisze określony ciąg znaków
2. Odczytać cały plik tekstowy linia po linii

3. Zapisać 10 losowych liczb do pliku binarnego
4. Odczytać cały plik binarny
5. Stworzyć histogram pliku binarnego
6. Wyznaczyć średnią wszystkich liczb zapisanych w pliku
7. Stworzyć klasę reprezentującą Osobę
8. Zapisać osoby do pliku
9. Odczytać osoby z pliku
10. Stworzyć łańcuch znaków na podstawie zawartości tablicy liczb
11. Odczytać z łańcucha znaków wszystkie liczby i zapisać je do kolekcji Vector