Politechnika Śląska w Gliwicach
Wydział Automatyki, Elektroniki i Informatyki

Podstawy Programowania Komputerów

D'Hondt

autor	Mateusz Piwowarski
prowadzący	dr inż. Krzysztof Simiński
rok akademicki	2017/2018
kierunek	informatyka
rodzaj studiów	SSI
semestr	1
termin laboratorium / ćwiczeń	wtorek, 13:45 – 15:15
grupa	3
sekcja	7
termin oddania sprawozdania	2017-12-01
data oddania sprawozdania	2017-12-01

1 Treść zadania

Napisać program wyznaczający liczbę mandatów uzyskanych w wyborach do parlamentu. W ordynacji wyborczej wykorzystywana jest metoda D'Hondta. Wyniki głosowania przekazywane w jednej linii pliku wejściowego: pierwszą wartością jest liczba miejsc w parlamencie, a następnie podawane są rozdzielone spacjami liczby głosów uzyskane przez partie. W pliku może znajdować się dowolna liczba przypadków (różne parlamenty, różne liczby głosów). Można przyjąć, że liczba partii nie jest większa od pewnego `MAX`. Do pliku wyjściowego zapisywane są liczby mandatów uzyskanych w wyborach. Nazwa pliku wejściowego podawana jest po przełączniku `-i`, wyjściowego po `-o`. Przykładowe wywołanie programu:

```
program.exe -i wejscie -o wyjscie
```

Jeżeli nie zostanie podany plik wejściowy, program czyta ze standardowego wejścia. Jeżeli nie zostanie podany plik wyjściowy, program pisze na standardowe wyjście.

2 Analiza zadania

Zagadnienie przedstawia problem podziału mandatów w wyborach parlamentarnych korzystając z ilości dostępnych mandatów oraz głosów oddanych na poszczególne partie. Przy podziale mandatów będzie wykorzystywana metoda D'Hondta.

2.1 Struktury danych

W programie wykorzystano tablice do przechowywania danych. Tablice przechowują informacje o ilości głosów oddanych na partie oraz informacje o ilości przyznanych mandatów dla każdej partii. Taka struktura danych umożliwia łatwe przechowywanie danych oraz porównywanie wartości.

2.2 Algorytmy

Metoda D'Hondta: Program porównuje wielkości ilorazów wyborczych. Iloraz wyborczy przyjmuje wartość według wzoru(1). Po porównaniu, przyznaje mandat partii z największym ilorazem wyborczym. Po przyznaniu mandatu, iloraz wyborczy zmniejsza się. Przy przydzielaniu następnego mandatu będą porównywane nowe ilorazy wyborcze. Program wykonuje tyle porównań ile jest możliwych mandatów do podziału.

Przykład działania algorytmu przedstawia Rysunek 1.

$$Iloraz\ wyborczy = \frac{ilość\ głosów\ uzyskanych\ w\ wyborach}{ilość\ mandatów\ uzyskanych\ +\ 1} \tag{1}$$

Mandat nr:	Czy partia 1 uzyskała mandat:	Ilorazy partii która uzyskała 720 głosów	Czy partia 2 uzyskała mandat:	Ilorazy partii która uzyskała 300 głosów	Czy partia 3 uzyskała mandat:	Ilorazy partii która uzyskała 480 głosów
1	TAK	720	NIE	300	NIE	480
2	NIE	360	NIE	300	TAK	480
3	TAK	360	NIE	300	NIE	240
4	NIE	240	TAK	300	NIE	240
5	TAK	240	NIE	150	NIE	240
6	NIE	180	NIE	150	TAK	240
7	TAK	180	NIE	150	NIE	160
8	NIE	144	NIE	150	TAK	160
	SUMA: 4		SUMA: 1		SUMA: 3	

Rysunek 1: Przykład podziału 8 mandatów metodą D'Hondta dla partii, które otrzymały kolejno: 720, 300, 480 głosów.

3 Specyfikacja zewnętrzna

Program jest uruchamiany z linii poleceń. Należy przekazać do programu nazwy plików (bez rozszerzenia): wejściowego i wyjściowego po odpowiednich przełącznikach (odpowiednio: `-i` dla pliku wejściowego i `-o` dla pliku wyjściowego) np.

```
program -i wejscie -o wyjscie
program -o wyjscie -i wejscie
```

Przełączniki mogą być podane w dowolnej kolejności. Uruchomienie programu z parametrem `-h` powoduje wyświetlenie krótkiej pomocy. Pomoc również jest wyświetlania w wyniku podania niepoprawnych danych.

```
program -h
```

Jeżeli nie zostanie podany przełącznik `-i` przy uruchomieniu programu lub zostanie podana nieprawidłowa nazwa pliku, dane potrzebne do wykonania programu będą pobierane z wejścia standardowego. Otrzymamy komunikat:

```
Nie podano lub nie znaleziono pliku wejsciowego.
```

Następnie program będzie wyświetlał instrukcje jakie dane należy podać:

```
Podaj liczbe partii (max. 100).
Podaj liczbe mandatow.
Podaj liczbe glosow na partie 1
Podaj liczbe glosow na partie 2
Podaj liczbe glosow na partie 3
Podaj liczbe glosow na partie n
```

Gdzie `n` w ostatnim z powyższych komunikatów oznacza numer kolejnych partii biorących udział w wyborach parlamentarnych.
Jeżeli nie zostanie podany przełącznik `-o` przy uruchamianiu programu, wynik programu będzie wyświetlany na wyjście standardowe:

```
Podzial mandatow:
partia 1: 4
partia 2: 1
partia 3: 3
partia n: k
```

Gdzie `n` w ostatnim z powyższych komunikatów oznacza numer kolejnych partii biorących udział w wyborach parlamentarnych. `k` oznacza ilość mandatów przyznanych dla partii `n`. Zapis wyniku do pliku wyjścia wygląda podobnie jak w przypadku wyświetlania wyniku na wyjście standardowe.

Przy wykorzystywaniu danych z pliku wejściowego: Pierwszą wartością jest liczba miejsc w parlamencie, a następnie liczby głosów uzyskane przez partie. Każda wartość powinna być oddzielona spacją oraz powinna być liczbą naturalną. Liczba podanych partii nie powinna przekraczać 100. Podanie niepoprawnych danych w pliku wywołuje komiunikat oraz kończy program:

```
Zle zapisane dane w pliku wejscia.
```

Podanie niepoprawnych danych w standardowym wejściu wywołuje komunikat oraz kończy program:

```
Wpisales bledne dane.
```

Podanie niepoprawnych danych wyświetli również instrukcję dla użytkownika.

4 Specyfikacja wewnętrzna

Program został zrealizowany zgodnie z paradygmatem strukturalnym. W programie rozdzielono interfejs (komunikację z użytkownikiem) od logiki aplikacji (podziału mandatów).

4.1 Typy zdefiniowane w programie

```
1 const int MAX = 100;
```

Typ ten służy do określenia największej możliwej ilości partii. Wartość `MAX` jest stała przez cały czas działania programu. Jest wykorzystywana w funkcjach:

`Pobieranie_danych , Podzial_mandatow`

oraz określa ilość elementów tablic:

`glosy[MAX] , iloscpartii[MAX]`

Zdefiniowano także typ tablic, które przechowują dane: ilość głosów otrzymanych przez każdą z partii oraz ilość mandatów uzyskanych przez partie. Wartości tablic są liczbami naturalnymi.

```
1 unsigned int glosy[MAX];
2 unsigned int mandaty[MAX];
```

Typ ten jest wykorzystywany przez funkcje:

`Wyzerowanie_mandatow , Wartosci_z_konsoli ,`
`Pobieranie_danych , Podzial_mandatow , Wynik_konsola , Wynik`

Zdefiniowano również typ zmiennych, które przechowują dane: ilość partii biorących udział w wyborach parlamentarnych oraz ilość mandatów do podziału. Wartości zmiennych są liczbami naturalnymi.

```
1 unsigned int iloscpartii;
2 unsigned int iloscmandatow;
```

Typ ten jest wykorzystywany przez funkcje:

`Poprawnosc_danych , Wyzerowanie_mandatow , Wartosci_z_konsoli,`
`Pobieranie_danych , Wynik_konsola , Wynik`

Zdefiniowano także typ klas string, które przechowują nazwy pliku wejścia i wyjścia.

```
string nazwa_pliku_wejscia;
string nazwa_pliku_wyjscia;
```

Typ ten jest wykorzystywany przez funkcje:

`Odczytaj_argumenty , Pobieranie_danych , Wynik`

4.2 Ogólna struktura programu

W funkcji głównej wywołana jest funkcja

```
void Odczytaj_argumenty(int ile, char ** argumenty,
    string & szInput, string & szOutput)
```

Funkcja ta sprawdza, czy program został wywołany wraz z użyciem przełączników `-i`, `-o`, `-h`. Jeżeli odpowiednie przełączniki zostały wykorzystane, funkcja w zmiennych szInput i szOutput przechowa nazwy pliku wejścia oraz wyjścia. Gdy program został wywołany z użyciem parametru `-h` Funkcja wywołuje funkcję:

```
void Pomoc();
```

Funkcja ta wyświetla instrukcję korzystania z programu.
Następnie sprawdzana jest wartość zwracana funkcji:

```
bool Pobieranie_danych(string wejscie, const int MAX,
    unsigned int glosy[], unsigned int & iloscpartii,
    unsigned int & iloscmandatow)
```

Funkcja ta otwiera i pobiera z pliku wejścia dane (plik umieszczony w katalogu zewnętrzynm. `dat`). Zostają przechowane dane w zmiennej i tablicy: `iloscmandatow`, `glosy[ ]`. Domyślnie zwracana jest wartość `True`. Jeżeli wystąpił błąd w danych odczytywanych z pliku wejścia zwracana jest wartość `False`. Gdy plik wejściowy nie istnieje lub jego nazwa nie jest podana, zostaje wywołana funkcja:

```
bool Wartosci_z_konsoli(unsigned int glosy[], unsigned
    int & iloscpartii, unsigned int & iloscmandatow)
```

Funkcja pobiera dane z standardowego wejścia. Zostają przechowane dane w zmiennych i tablicy: `iloscpartii`, `iloscmandatow`, `glosy[ ]`. Domyślnie zwracana jest wartość `True`. Po otrzymaniu każdej wartości zostaje uruchomiona funkcja

`Poprawnosc_danych`

, która sprawdza czy dane podane przez użytkownika są poprawne. Jeżeli nie są, zostaje zwrócona wartość `False`.
Wywołana wcześniej funkcja

```
1 void Poprawnosc_danych(bool & poprawne_dane, const
    unsigned int iloscpartii)
```

sprawdza poprawność wpisanych wartości przez użytkownika. Jeżeli wartości są niepoprawne ustawia wartość `niepoprawnedane` na `False` oraz wywołuje funkcję `Pomoc()`.

Jeżeli w funkcji głównej funkcja

`Pobieranie_danych`

zwróciła wartość False program zostaje zakończony. W przypadku wartości True zostaje wywołana funkcja

```
1 void Wyzerowanie_mandatow(unsigned int mandaty[ ],
    unsigned int iloscpartii)
```

Funkcja ustawia wszystkie wartosci tablicy `mandaty[ ]` na 0.

Następną wywołaną funkcją w funkcji głównej jest

```
1 void Podzial_mandatow(const int MAX, unsigned int
    mandaty [ ], unsigned int glosy [ ], unsigned int
    iloscpartii, unsigned int iloscmandatow)
```

Funkcja zmienia wartości tablicy mandatów dla każdej partii. Tworzy kopię tablicy głosów, następnie porównując każdą z wartości przyznaje mandat dla tego samego miejsca w tablicy mandatów, gdzie była najwyższa wartość w tablicy kopii głosów. Po przyznaniu mandatu, w tym samym miejscu w kopii tablicy pojawia się nowa wartość, która jest ustawiana przez funkcję

```
1 double lloraz_wyborczy(unsigned int glosy, unsigned int
    mandaty)
```

Funkcja zwraca nową wartość w kopii tablicy głosów, która zdobyła mandat. Wartość zwracana jest zależna od wartości w tablicy głosów dla danej partii oraz ilość mandatów która jest im przyznana.

Następnie w funkcji głównej zostaje wywołana funkcja

```
1 void Wynik(string wyjscie, unsigned int iloscpartii,
    unsigned int mandaty[ ])
```

Funkcja sprawdza czy nazwa pliku wyjścia jest pusta. Jeżeli nazwa pliku wyjścia nie jest pusta, zostaje utworzony plik tekstowy w zewnętrznym katalogu `dat` oraz zostaje tam zapisany wynik podziału mandatów. Funkcja korzysta z wartości znajdujących się w tablicy `mandaty`. Jeżeli nazwa pliku wyjścia jest pusta, zostaje wywołana funkcja

```
1 void Wynik_konsola(unsigned int mandaty[ ], unsigned
    int iloscpartii)
```

Funkcja wyswietla w standardowym wyjściu wynik podziału mandatów. Funkcja korzysta z wartości znajdujących się w tablicy `mandaty`.

4.3 Szczegółowy opis implementacji funkcji

```
1 void Odczytaj_argumenty(int ile, char ** argumenty,
    string & szInput, string & szOutput)
```

Funkcja ta sprawdza, czy program został wywołany wraz z użyciem przełączników `-i`, `-o`, `-h`. Parametry funkcji:
`ile` – ilość parametrów wywołanych w wierszu linii poleceń. `argumenty` – tablica parametrów wywołanych w wierszu linii poleceń. `szInput` – zmienna typu string w której znajduje się nazwa pliku wejścia. `szOutput` – zmienna typu string w której znajduje się nazwa pliku wyjścia

Wewnatrz funkcji są zadeklarowane zmienne typu string przechowujące nazwę przełączników:
`ETYKIETAINPUT(-i")` – przechowuje przełącznik `-i`;
`ETYKIETOUTPUT(-o")` – przechowuje przełącznik `-o`;
`ETYKIETAINPUT(-h")` – przechowuje przełącznik `-h`;
Funkcja używa pętli sprawdzając każdy argument w wierszu linii poleceń. W tym celu jest stworzona zmienna `arg`, która jest aktualnym elementem argumentów w pętli. Wewnątrz pętli zmienna `arg` jest porównywana do każdej zmiennej która przechowuje przełącznik. Pętla sprawdza argumenty od miejsca pierwszego tablicy, ponieważ miejscem zerowym jest nazwa programu, która nie może być przełącznikiem. Ilość wykonanych pętli zależy od zmiennej `ile`, która jest parametrem funkcji. Jeżeli zmienna `arg` ma taką samą wartość jak wartość zmiennej `ETYKIETAINPUT`: wartość następnego argumentu w wierszu linii poleceń jest zapisana do zmiennej `szInput`. Jeżeli zmienna `arg` ma taką samą wartość jak wartość zmiennej `ETYKIETAOUTPUT`: wartość następnego argumentu w wierszu linii poleceń jest zapisana do zmiennej `szOutput`. Jeżeli zmienna `arg` ma taką samą

wartość jak wartość zmiennej ETYKIETAHELP: wywoływana jest funkcja Pomoc.

```
1 void Pomoc();
```

Funkcja wyświetla instrukcję korzystania z programu. W instrukcji jest opisane w jaki sposób powinien być wywołany program oraz jakie dane powinny być zamieszczone w pliku wejścia lub w danych podanych przez użytkownika.

```
1 bool Pobieranie_danych(string wejscie, const int MAX,
    unsigned int glosy[ ], unsigned int & iloscpartii,
    unsigned int & iloscmandatow)
```

Parametry funkcji:
wejscie – zmienna typu string, w której znajduje się nazwa pliku wejścia. MAX – zmienna określająca maksymalną ilość partii biorących udział w wyborach parlamentarnych. glosy [ ] – tablica przechowująca ilości głosów otrzymanych przez partie. iloscpartii – zmienna przechowująca ilość partii biorących udział w wyborach parlamentarnych. iloscmandatow – zmienna przechowująca ilość mandatów, które są podzielone w wyborach parlamentarnych

Funkcja zwraca informacje czy pobrane wartości z pliku wejścia lub z standardowego wejścia są poprawne. True – wartości poprawne False – wartości niepoprawne. Informacja o poprawności danych (wartość zwracana) jest przypisana zmiennej poprawnedane

Funkcja tworzy strumień wejścia do pliku tekstowego w katalogu zewnętrznym dat wykorzystując nazwę pliku w zmiennej wejscie. Następnie funkcja sprawdza czy plik istnieje oraz czy jest możliwy dostęp. Gdy dostęp jest możliwy, funkcja zapisuje dane z pliku do zmiennej iloscmandatow oraz do tablicy glosy[ ]. Zapis danych w tablicy jest wywoływany pętlą która się kończy gdy dojdziemy do końca pliku lub gdy pobierzemy ilość głosów dla 100 partii. Jeżeli dane w pliku są zapisane niepoprawnie, funkcja zmienia wartość poprawnedane na False, wyświetla komunikat oraz wywołuje funkcję Pomoc(). Po wykonaniu wszystkich operacji na otwartym pliku, funkcja zamyka plik.
Jeżeli funkcja nie znajdzie pliku wejściowego, sprawdza wartość zwracaną funkcji Wartosc z konsoli. Jeżeli funkcja Wartosc z konsoli zwraca wartość False, zmienna poprawnedane zmieniają swoją wartość na False.

```
1 bool Wartosci_z_konsoli(unsigned int glosy[ ], unsigned
    int & iloscpartii, unsigned int & iloscmandatow)
```

Parametry funkcji:
`glosy [ ]` – tablica przechowująca ilości głosów otrzymanych przez partie. `iloscpartii` – zmienna przechowująca ilość partii biorących udział w wyborach parlamentarnych. `iloscmandatow` – zmienna przechowująca ilość mandatów, które są podzielone w wyborach parlamentarnych

Funkcja zwraca informacje czy pobrane wartości z standardowego wejścia są poprawne. `True` – wartości poprawne `False` – wartości niepoprawne. Informacja o poprawności danych (wartość zwracana) jest przypisana zmiennej `poprawnedane`. Domyślnie zmienna `poprawnedane` przyjmuje wartość `True`

Funkcja wyświetla komunikaty, jakie wartości powinien wprowadzić użytkownik. Po każdym pobraniu wartości od użytkownika, program wywołuje funkcje Poprawnosc danych, która sprawdza czy wartości są poprawne.
Funkcja przypisuje dane podane przez użytkownika do zmiennych `iloscpartii`, `iloscmandatow` oraz do tablicy `glosy`.

```
1 void Poprawnosc_danych(bool & poprawne_dane, const
    unsigned int iloscpartii)
```

Parametry funkcji:
`poprawne dane` – zmienna przechowująca informacje o poprawności danych `iloscpartii` – zmienna przechowująca ilość partii biorących udział w wyborach parlamentarnych
Funkcja sprawdza czy wartości wprowadzone przez użytkownika są poprawne oraz czy użytkownik nie podał większej ilości partii niż 100. Jeżeli dane wprowadzone są niepoprawne, funkcja wyświetla komunikat, zmienia wartość zmiennej `poprawne dane` na `False` oraz wywołuje funkcję `Pomoc()`

```
1 void Wyzerowanie_mandatow(unsigned int mandaty[ ],
    unsigned int iloscpartii)
```

Parametry funkcji:
`mandaty [ ]` – tablica przechowująca dane o ilości otrzymanych mandatów przez partie
`iloscpartii` – zmienna przechowująca ilość partii biorących udział w wyborach parlamentarnych
Funkcja wywołuje pętlę która ustawia liczbę wszystkich mandatów otrzymanych przez partię na 0. Ilość powtórzeń pętli jest zależna od wartości zmiennej `iloscpartii`.

```
1 void Podzial_mandatow(const int MAX, unsigned int
    mandaty [ ], unsigned int glosy [ ], unsigned int
    iloscpartii , unsigned int iloscmandatow)
```

Parametry funkcji:
`MAX` – zmienna określająca maksymalną ilość partii biorących udział w wyborach parlamentarnych
`mandaty [ ]` – tablica przechowująca dane o ilości otrzymanych mandatów przez partie
`glosy [ ]` – tablica przechowująca ilości głosów otrzymanych przez partie `iloscpartii` – zmienna przechowująca ilość partii biorących udział w wyborach parlamentarnych `iloscmandatow` – zmienna przechowująca ilość mandatów, które są podzielone w wyborach parlamentarnych

Funkcja tworzy tablicę `baza glosow`. Wartości `baza glosow` zostają ustawione na wartości z tablicy textttglosy. W funkcji zostają również stworzone zmienne textttmiejscemax oraz textttmaks. Te zmienne pomagają w porównywaniu ilorazów wyborczych w `baza glosow`. textttmiejscemax przechowuje miejsce tablicy w ktorej jest najwiekszy iloraz wyborczy a textttmaks przechowuje najwyższą wartość ilorazów wyborczych. Przyjmują kolejno wartości 0 oraz -1 aby pierwszy element porównywany jednocześnie był miejscem tablicy w którym znajduje się największy iloraz wyborczy i którego wartość ilorazu wyborczego była najwyższa.
Po porównaniu każdego elementu tablicy textttbaza glosow, w miejscu gdzie była największa wartość ilorazu wyborczego w tym samym miejscu tablicy textttmandaty zwiększa się wartość o 1. Oznacza to przyznanie jednego mandatu dla partii o najwyższym ilorazie wyborczym. Po przyznaniu mandatu, w miejscu textttmiejscemax tablicy textttbaza glosow przypisujemy nową wartość zwracaną przez funkcję textttIloraz wyborczy. Procedura przypisania mandatu się powtarza, aż nie zostaną wyznaczone wszystkie mandaty dla partii.

```
1 double Iloraz_wyborczy(unsigned int glosy , unsigned int
    mandaty)
```

Parametry funkcji:
`glosy` – zmienna określająca ilość głosów otrzymanych przez partię wyborach `mandaty` – zmienna określająca ilość mandatów otrzymanych przez partię
Funkcja zwraca nową wartość ilorazu wyborczego partii, która zdobyła

mandat. Wartość zwracana jest zależna od wartości w tablicy głosów dla danej partii oraz ilość mandatów która jest im przyznana.

```
1 void Wynik(string wyjscie, unsigned int iloscpartii,
    unsigned int mandaty[])
```

Parametry funkcji:
`wyjscie` – zmienna typu string, w której znajduje się nazwa pliku wyjscia
`iloscpartii` – zmienna przechowująca ilość partii biorących udział w wyborach parlamentarnych
`mandaty [ ]` – tablica przechowująca dane o ilości otrzymanych mandatów przez partie

Funkcja sprawdza czy zmienna `wyjscie` jest pusta. Jeżeli jest pusta, zostaje wywołana funkcja `Wynik konsola`. Jeżeli jest inaczej, funkcja tworzy strumień wyjścia do pliku tekstowego w katalogu zewnętrznym `dat` wykorzystując nazwę pliku w zmiennej `wyjscie`. Następnie zostaje zapisany wynik w pliku tekstowym korzystając z danych przechowanych w tablicy `mandaty`. Wynik ukazuje kolejne partie wraz z ilością uzyskanych mandatów. Rezultat dla każdej partii jest zapisany jeden pod drugim. Po zapisaniu wyniku w pliku wyjściowym, funkcja zamyka plik.

```
1 void Wynik_konsola(unsigned int mandaty[], unsigned
    int iloscpartii)
```

Parametry funkcji:
`mandaty [ ]` – tablica przechowująca dane o ilości otrzymanych mandatów przez partie
`iloscpartii` – zmienna przechowująca ilość partii biorących udział w wyborach parlamentarnych

Funkcja wyswietla w standardowym wyjściu wynik podziału mandatów. Funkcja korzysta z wartości znajdujących się w tablicy `mandaty`. Wynik przedstawia kolejnne partie wraz z ilością uzyskanych mandatów. Rezultat dla każdej z partii jest zapisany jeden pod drugim.

5 Testowanie

Program został przetestowany na różnego rodzaju plikach. Pliki niepoprawne (niezawierające liczb, zawierające liczby zmiennoprzecinkowe, liczby ujemne oraz znaki, które nie są liczbami.) powodują zgłoszenie błędu. Plik pusty również powoduje zgłoszenie błędu. W pliku `zle1.txt` znajdują

się litery zamiast liczb. Plik `zle2.txt` przechowuje wartości ujemne. W pliku `zle3.txt` znajdują się liczby zmiennoprzecinkowe. Program został sprawdzony pod kątem wycieków pamięci. Plik `zle4.txt` jest pustym plikiem.
Pliki poprawne przechowują poprawne dane. Pliki, w którym znajdują się dodatkowe znaki białe między wartościami nie powodują zgłoszenia błędu przez program. Trywialny wynik otrzymujemy po wczytaniu pliku `wejscie.txt` lub `wejscie1.txt`, ponieważ przechowują proste dane zapisane w poprawny sposób.
W zbiorach testowych również pojawiły się pliki poprawne, ale nietypowe. Plik `wejscie2.txt` nie powoduje zgłoszenia błędu, mimo większych odstępów między danymi. Plik `wejscie3.txt` przechowuje duże wartości. Podzielenie mandatów dla danych w pliku `wejscie3.txt` mogłoby stwarzać problemy lub zajmować sporo czasu, jednak program zapisuje poprawne dane. Metoda D'Hondta nie przewiduje rozwiązania dla przypadku, w którym partie otrzymały jednakową ilość głosów. Program w takiej sytuacji przydziela mandat partii, która jest wyżej na liście. Przykład takiego rozwiązania otrzymamy przy wykorzystaniu pliku `wejscie4.txt`.
Przykład wyniku dla pliku poprawnego:

```
Podzial mandatow:
partia 1: 4
partia 2: 1
partia 3: 3
```

Uruchomienie programu z wykorzystaniem pliku niepoprawnego wyświetla komunikat: `Zle zapisane dane w pliku wejscia.` oraz wyświetla instrukcję obsługi programu.

6 Wnioski

Program do podziału mandatów metodą D'Hondta jest programem prostym, chociaż wymaga zaznajomienia się z teorią Metody D'Hondta. Najbardziej wymagające okazało się zapobieganie przed niepoprawnymi danymi. Realizacja projektu nauczyła podstawowych funkcji przy obsługach plików oraz praktycznego pisania funkcji, tak aby kod był czytelny a funkcja nie zajmowała dużej ilości kodu.