Zajęcia 6 – podstawy obsługi plików

1. Napisać funkcję, która w pliku tekstowym o podanej nazwie zlicza:

   - liczbę wszystkich znaków w pliku,
   - liczbę białych znaków w pliku (spacja, tabulator, itd.),
   - liczbę słów – przez słowo rozumiemy dowolny ciąg niezawierający *białych znaków*,
   - liczbę wierszy.

   Funkcja w standardowy sposób zwraca liczbę znaków w pliku, natomiast liczba białych znaków słów i wierszy przekazywana jest przez parametry.

   ```
   int licz(const char nazwaPliku[],
            int &biale_znaki, int &slowa, int &wiersze)
   ```

   Przykład użycia:

   ```
   int main() {
       int biale_znaki, slowa, wiersze;
       int wszystkie = licz("zad1.txt", biale_znaki, slowa, wiersze);
       cout << "Wszystkie znaki: " << wszystkie
            << "\nBiale znaki: " << biale_znaki
            << "\nSlowa: " << slowa
            << "\nWiersze: " << endl;
       return 0;
   }
   ```

   Dla pliku „zad1.txt”:
   ␣␣Ala␣ma ↵
   ↵
   kota␣i␣3␣psy.

   Program powinien wyświetlić:

   ```
   Wszystkie znaki: 25
   Biale znaki: 10
   Slowa: 6
   Wiersze: 3
   ```

2. Napisać funkcję:

```
void szukaj(const char nazwaPlikWe[], const char nazwaPlikWy[],
            const char slowo[])
```

której zadaniem jest znalezienie wszystkich wierszy w pliku, które *zawierają* szukane słowo. Wszystkie wiersze, które zawierają słowo powinny zostać zapisane w pliku wynikowym wraz z nr wiersza (z pierwszego pliku). Nazwa pierwszego pliku zapamiętana jest w parametrze `nazwaPlikWe`, nazwa pliku wynikowego podana jest w parametrze `nazwaPlikWy`, natomiast szukane słowo w parametrze `slowo`.

Przykład - plik wejściowy:

```
Ala ma jutro egzamin z biologii.
Jan myje auto.
Eh, jutro kolejny egzamin.
Nie lubie polityki.
```

Jeżeli szukanym słowem byłoby "egzamin", to plik wynikowy powinien wyglądać następująco:

```
1: Ala ma jutro egzamin z biologii.
3: Eh, jutro kolejny egzamin.
```

3. Napisać funkcję `emerytura(char nazwaPliku[])`, która wczyta z pliku o podanej nazwie dane pracowników zapisane w kolejnych wierszach w następujący sposób:

`Imię Nazwisko Płeć Wiek`

Przykład:

```
Tomasz Nowak M 45
Marta Ziobro K 42
Jan Kowalski M 27
Ewelina Tusk K 59
```

Następnie funkcja dla każdego pracownika powinna wyznaczyć ile lat pozostało do jego emerytury. Wyniki należy zapisać w następujący sposób: Nazwisko Imię "Lata do emerytury"

Przykład:

```
Nowak Tomasz 20
Kowalski Jan 38
```

Wyniki dla kobiet należy zapisać w pliku o nazwie „kobiety.txt", natomiast wyniki dla mężczyzn należy zapisać w pliku „mezczyzni.txt".

4. Napisać funkcję `void sumuj_i_zapisz(const char* nazwa_pliku)`, która odczytuje plik o podanej nazwie zawierający liczby całkowite. Funkcja ta ma za zadanie dopisać w nowym wierszu na końcu tego pliku sumę odczytanych liczb powiększoną o 1, tak więc ponowne uruchomienia funkcji będą skutkowały dopisywaniem kolejnych wierszy. Jeżeli plik nie istnieje to ma zostać utworzony. *Uwaga.* Suma dla pustego pliku wynosi 0.

5. Zadanie polega na stworzeniu dwóch funkcji:

   ```
   void szyfruj(string nazwaPlikuWe, int przesun)
   void deszyfruj(string nazwaPlikuWe, int przesun)
   ```

   Funkcja szyfruj dokonuje szyfrowania pliku, którego nazwa podana została jako pierwszy parametr. Szyfrowanie polega na zamianie każdej litery na znak przesunięty o wartość podaną drugim parametrem np. dla przesunięcia równego 2 literka 'a' powinna zostać zastąpiona literką 'c', literka 'z' literką 'b' itp.

   Wynikiem działania funkcji ma być plik o nazwie utworzonej na podstawie nazwy pliku wejściowego poprzez dołączenie znaku '_' np. dla pliku dane.txt zaszyfrowana postać powinna mieć nazwę `_dane.txt`. Funkcja deszyfruj powinna deszyfrować plik (niekoniecznie ten sam) zaszyfrowany przez funkcję szyfruj.

6. Napisać program, którego zadaniem jest odczytanie danych tabelarycznych w pliku tekstowym, a następnie zapisanie ich do nowego pliku w postaci kodu HTML. Wczytane dane tabelaryczne mają zostać umieszczone w tabeli HTML. Uwaga, liczba kolumn nie jest zadana z góry, jednak można założyć, że jest identyczna w każdym z wierszy. Za separator kolejnych wartości należy przyjąć spację.

   Przykładowo, dla pliku zawierającego:

   ```
   "Waga" "Wzrost" "BMI" "Nadwaga"
   70 1,8 21,6 "NIE"
   67 1,77 21,39 "NIE"
   85 1,7 29,41 "TAK"
   100 1,92 27,13 "TAK"
   ```

   wynikiem powinien być plik HTML o następującej treści:

   ```
   <html><body>
   <table>
   <tr><td>"Waga"</td><td>"Wzrost"</td><td>"BMI"</td><td>"Nadwaga"
   </td></tr>
   <tr><td>70</td><td>1,8</td><td>21,6</td><td>"NIE"
   </td></tr>
   <tr><td>67</td><td>1,77</td><td>21,39</td><td>"NIE"
   </td></tr>
   <tr><td>85</td><td>1,7</td><td>29,41</td><td>"TAK"
   ```

```
</td></tr>
<tr><td>100</td><td>1,92</td><td>27,13</td><td>"TAK"</td></tr>
</table>
</body></html>
```

7. Napisać program, który dla pliku tekstowego o podanej nazwie wyznaczy „wykres" częstości wystąpień małych liter alfabetu angielskiego. Słupki wykresu mają zostać utworzone ze znaków gwiazdki '*', przy czym długość słupka dla najczęściej występującej litery powinna wynosić 50. Dodatkowo dla każdego znaku należy dodatkowo wyświetlić liczbę jego wystąpień.

   Poniżej umieszczono przykładowy wykres wygenerowany dla tekstu „Adventures of Huckleberry Finn" M. Twaina dostępnego pod adresem:
   `http://www.gutenberg.org/dirs/7/76/76.txt`

```
a ************************************              36581
b *******                                          7439
c ********                                         8317
d ************************                         23754
e ************************************************* 49084
f ********                                         7914
g **********                                       10733
h *************************                        26338
i ****************************                     28222
j *                                                1211
k *****                                            5677
l ****************                                 17446
m **********                                       10337
n *********************************                32818
o ************************************             36700
p ******                                           5971
q                                                  189
r ********************                             20252
s ************************                         25193
t ******************************************       42390
u **************                                   13954
v **                                               2944
w *************                                    13347
x                                                  453
y **********                                       10312
z                                                  185
```