Zad 1

Napisz program, który sprawdza czy numer PESEL jest poprawny. Numery PESEL mają być czytane z pliku `pesele.txt`. Każdy numer jest w oddzielnej linii. W wyniku program ma zwrócić informację, ile było poprawnych numerów PESEL w pliku. Algorytm sprawdzania jest następujący:

4	4	0	5	1	4	0	1	3	5	8
a	b	c	d	e	f	g	h	i	j	k

PESEL jest poprawny, jeśli:
*(10-K) = (1*a + 3*b + 7*c + 9*d + 1*e + 3*f + 7*g + 9*h + 1*i + 3*j) (mod 10)*
Przykład dla numeru PESEL 44051401358:
1*4 + 3*4 + 7*0 + 9*5 + 1*1 + 3*4 + 7*0 + 9*1 + 1*3 + 3*5 = 101
Wyznaczamy resztę z dzielenia sumy przez 10: 101/10 = 10 reszta = 1
Jeżeli reszta = 0, to cyfra kontrolna wynosi 0. Jeżeli reszta ≠ 0, to cyfra kontrolna będzie uzupełnieniem reszty do 10, czyli w podanym przykładzie jest to cyfra 9.
10 - 1 = 9
Wynik 9 nie jest równy ostatniej cyfrze numeru PESEL, czyli 8, więc numer zawiera błąd.

Zad 2

Dana jest następująca struktura w pliku tekstowym

```
nazwisko; ocena_dyplom; czyWnioslOplate
```

przykładowo:

Kowalski;5.0;tak

Napisz funkcję `doObrony(string nazwaPliku)`:

a) Funkcja zwraca liczbę całkowitą studentów dopuszczonych do obrony (takich, którzy mają ocenę na dyplomie > 3 oraz wniesioną opłatę (`czyWnioslOplate` równe „tak“).

b) Funkcja powinna zapisywać do pliku tekstowego `<nazwaPliku>` w kolejności alfabetycznej nazwiska studentów dopuszczonych do obrony.

Zad 3

Napisz funkcję `liczbaRozdzialow(string nazwaPliku, bool & ok)`, która przyjmuje nazwę pliku tekstowego oraz:

1. Zwraca liczbę całkowitą wystąpień słowa *„rozdział”* w tekście
2. W zmiennej `ok` ustawia wartość, czy numeracja rozdziałów jest ciągła (czyli nie ma „dziur”)
3. Uwzględnij, iż słówko *rozdział* może być pisane wielkimi lub małymi literami (lub mieszanie).

Zad 4

Napisz funkcję `generujTozsamosc(int ile, string[] imiona, string[] nazwiska, int[] numery)`, która:

a) Generuje *ile* danych osób (imię, nazwisko, nr telefonu) i zapisującą je do pliku tekstowego *daneXXX.txt*, gdzie XXX – to liczba *ile* (zapisana na **trzech** pozycjach).

b) Dane osób są losowane z tablic `imiona`, `nazwiska` oraz `numery`, przy czym każdą z danych można wykorzystać tylko **raz**.

c) Jeśli braknie danych służących do generowania, funkcja powinna zwrócić **false**.

Zad 5

Napisać procedurę `void emerytura(String nazwaPliku)`, która wczyta z pliku o podanej nazwie dane pracowników zapisane w kolejnych wierszach w następujący sposób:

```
Imię Nazwisko Płeć Wiek
```

Pola imię i nazwisko to ciągi znaków, płeć to jeden znak, natomiast wiek to liczba całkowita. Następnie procedura dla każdego pracownika powinna wyznaczyć, ile lat zostało do jego emerytury (zakładając, że wiek emerytalny mężczyzn to 65, a kobiet 60 lat). Wyniki należy zapisać w pliku *mezczyzni.txt* lub *kobiety.txt* (w zależności od płci pracownika) w następujący sposób:

```
Nazwisko Imię „Lata do emerytury”
```

Przykład:

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

Plik mezczyzni.txt:

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

Plik kobiety.txt

```
Tusk Ewelina 1
Ziobro Marta 18
```

Zad 6

Napisać procedurę `void szukaj(String plikWe, String plikWy, String slowo)`, której zadaniem jest znalezienie wszystkich wierszy w pliku, które zawierają szukane słowo. Wszystkie wiersze, które zawierają szukane słowo powinny być zapisane w pliku wynikowym wraz z nr wiersza (z pierwszego pliku). Nazwa pierwszego pliku zapamiętywana jest w parametrze `plikWe`, nazwa pliku wynikowego podana jest w parametrze `plikWy`, 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 wyjściowy powinien wyglądać następująco:

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

Zad 7

Napisać funkcję `przepisz`, która jako pierwszy parametr otrzymuje nazwę pliku tekstowego, w którym każda linia wygląda następująco:

```
imię*nazwisko*plec*wiek*pensja
```

gdzie imię i nazwisko zapisane są literami alfabetu angielskiego, płeć to litera *'K'* lub *'M'*, a wiek i pensja, to liczby całkowite dodatnie. Liczba osób zapisanych w pliku jest nieokreślona. W rezultacie jej działania powinny powstać dwa pliki wyjściowe, gdzie jeden będzie zawierał jedynie kobiety (nazwa taka sama jak pliku w przypadku wejściowego, poprzedzona literą *'k'*), a drugi mężczyzn (nazwa poprzedzona literą *'m'*).

Jednocześnie do nowych plików należy:

- nie przepisywać oznaczenia płci,
- kobietom o wieku większym niż podany jako drugi parametr (wiek) podnieść pensję o 15%,
- mężczyznom podnieść pensję o tyle, ile mają lat.

W wyniku działania funkcji powinien zostać zwrócony średni wiek wszystkich mężczyzn z pliku wejściowego.

Zad 8

Informatyk z firmy „KompOK” zapisał w pliku *hasla.txt* 200 haseł. Każde hasło umieszczone jest w osobnym wierszu pliku. Hasło składa się tylko z małych liter alfabetu angielskiego, zaś jego długość wynosi od 3 do 10 znaków. Wykorzystując dane zawarte w tym pliku, wykonaj poniższe polecenia. Odpowiedzi do poszczególnych podpunktów zapisz w pliku tekstowym *odpowiedzi1.txt* opatrując je numerem podpunktu:

1. Podaj, ile haseł ma parzystą, a ile nieparzystą liczbę znaków.
2. Utwórz zestawienie haseł (po jednym w wierszu), które są palindromami. Palindrom to wyraz brzmiący tak samo przy czytaniu z lewej strony do prawej, jak i odwrotnie, np. kajak, potop.
3. Utwórz zestawienie haseł (po jednym w wierszu) zawierających w sobie dwa kolejne znaki, których suma kodów ASCII wynosi 220.

Przykłady:

* Hasło *krzysio* zawiera dwa kolejne znaki *si*, których suma kodów ASCII wynosi 220. Kod ASCII znaku *s* to 115, kod znaku *i* to 105; suma kodów wynosi 115+105 = 220.
* Hasło *cyrk* zawiera również takie dwa kolejne znaki. Kod ASCII znaku *c* to 99, kod ASCII znaku *y* to 121; suma kodów wynosi 99+121=220.

Zad 9

Napisz program „Jednoręki bandyta”. Na początku gracz posiada 100 żetonów. Każde pociągnięcie dźwigni jednorękiego bandyty odejmuje mu z konta 5 żetonów. Po pociągnięciu losowanych jest 9 liczb w następującej konfiguracji:

L1	L2	L3
L4	L5	L6
L7	L8	L9

Gracz wygrywa, jeśli w którymkolwiek poziomie, pionie lub ukosie znajdą się te same liczby. Wygrana jest uzależniona od rodzaju liczb: stawka (5 żetonów) * 2 * liczba.

Program ma umożliwiać zakończenie gry w dowolnym momencie.