Algorytm liczący dni tygodnia - metoda konwergencji Zellera

Treść zadania:

Napisz program w języku C#, który obliczy dzień tygodnia dla podanej przez użytkownika daty na podstawie metody Zellera. Program powinien spełniać następujące wymagania:

1. Użytkownik podaje datę w formacie dd.MM.yyyy (np. 29.09.2024).
2. Program przetwarza wprowadzone dane i oblicza dzień tygodnia za pomocą poniższego wzoru Zellera:

Zgodność Zellera

Kongruencja Zellera to algorytm opracowany przez Christiana Zellera w celu **obliczenia dnia tygodnia dla dowolnej daty kalendarza juliańskiego lub gregoriańskiego** . Można go uznać za oparty na konwersji między dniem juliańskim a datą kalendarzową.
Jest to algorytm służący do znajdowania dnia tygodnia dla dowolnej daty.
W przypadku kalendarza gregoriańskiego jest to: W przypadku kalendarza juliańskiego jest to: gdzie,

---

For the Gregorian calendar, Zeller's congruence is

$$h = \left(q + \left\lfloor \frac{13(m+1)}{5} \right\rfloor + K + \left\lfloor \frac{K}{4} \right\rfloor + \left\lfloor \frac{J}{4} \right\rfloor - 2J\right) \mod 7,$$

for the Julian calendar it is

$$h = \left(q + \left\lfloor \frac{13(m+1)}{5} \right\rfloor + K + \left\lfloor \frac{K}{4} \right\rfloor + 5 - J\right) \mod 7,$$

1. h to dzień tygodnia (0 = sobota, 1 = niedziela, 2 = poniedziałek, ..., 6 = piątek)
2. q to dzień miesiąca
3. m to miesiąc (3 = marzec, 4 = kwiecień, 5 = maj, ..., 14 = luty)
4. K jest rokiem stulecia (rok % 100).
5. J to stulecie zerowe (właściwie ? rok/100 ?). Na przykład stulecia zerowe dla lat 1995 i 2000 to odpowiednio 19 i 20 (nie należy ich mylić ze zwykłym porządkowym wyliczeniem stuleci, które w obu przypadkach wskazuje 20. wiek).

```
UWAGA: W tym algorytmie styczeń i luty
      liczone są jako miesiące 13. i 14. poprzedniego
      roku. Np. jeśli jest 2 lutego 2010 r.,
      algorytm uznaje tę datę za drugi dzień
      czternastego miesiąca 2009 r. (02/14/2009
      w formacie DD/MM/RRRR).
```

W przypadku daty tygodnia ISO Dzień tygodnia d (od 1 = poniedziałek do 7 = niedziela) użyj

```
d = ((h+5)%7) + 1
```

Examples [edit]

For 1 January 2000, the date would be treated as the 13th month of 1999, so the values would be:

$q = 1$
$m = 13$
$K = 99$
$J = 19$

So the formula evaluates as $(1 + 36 + 99 + 24 + 4 - 38) \bmod 7 = 126 \bmod 7 = 0 = \text{Saturday}$.

(The 36 comes from $(13 + 1) \times 13/5 = 182/5$, truncated to an integer.)

However, for 1 March 2000, the date is treated as the 3rd month of 2000, so the values become

$q = 1$
$m = 3$
$K = 0$
$J = 20$

so the formula evaluates as $(1 + 10 + 0 + 0 + 5 - 40) \bmod 7 = -24 \bmod 7 = 4 = \text{Wednesday}$.

3. Program powinien zawierać walidację wejścia:
    - Sprawdzenie, czy użytkownik podał datę w formacie dd.MM.yyyy.
    - Sprawdzenie poprawności wartości dla dnia i miesiąca (np. luty nie może mieć więcej niż 29 dni w roku przestępnym i 28 dni w roku nieprzestępnym).
4. Program powinien korzystać z obiektowego podejścia:
    - Stwórz klasę DateCalculator, która będzie zawierała:
        - Konstruktor przyjmujący datę w formie string.
        - Metody pomocnicze do obliczenia dnia tygodnia oraz do walidacji daty.
        - Metodę IsLeapYear(int year), która oblicza, czy dany rok jest przestępny. Rok jest przestępny, jeśli spełnia jeden z następujących warunków:
            - Rok jest podzielny przez 4, ale nie przez 100.
            - Rok jest podzielny przez 400.

- W klasie Program w metodzie Main wywołaj metodę klasy DateCalculator, IsLeapYear aby uzyskać wynik.

5. Program powinien wyświetlić wynik w formie tekstowej, np.: "29.09.2024 to niedziela" oraz „2024- jest rokiem przystępnym.”
6. Program musi przestrzegać konwencji nazewnictwa zmiennych:
    - Nazwy zmiennych muszą być rzeczowe, zgodne z konwencją.
    - Nazwy metod muszą być zgodne z konwencją.
7. **Dokumentacja aplikacji:**
    - Utwórz dokumentację do aplikacji w formie komentarza nad klasami DateCalculator oraz Program. Użyj poniższego wzoru, uzupełniając odpowiednie informacje:

****

klasa: <nazwa klasy>

opis: <co klasa reprezentuje i wykonuje?> metody: <nazwa metody1 - co zwraca>

<nazwa metody> - co zwraca> autor: <numer zdającego>

****

8. **Zrzuty ekranu:**
    - Wykonaj zrzuty ekranu dokumentujące uruchomienie aplikacji:
        - Zrzuty powinny obejmować cały obszar ekranu monitora z widocznym paskiem zadań.
        - Jeżeli aplikacja uruchamia się poprawnie, na zrzucie powinno być widoczne okno z wynikiem działania programu oraz otwarte środowisko programistyczne z projektem lub okno terminala z kompilacją projektu.
        - Jeżeli aplikacja nie uruchamia się z powodu błędów kompilacji, na zrzucie powinno być widoczne okno ze spisem błędów oraz otwarte środowisko programistyczne.
    - Wymagana liczba zrzutów zależy od liczby interakcji programu z użytkownikiem. Zapisz zrzuty jako konsola1, konsola2, ... itd.