Kurs .NET Tomasz Bursztyński

ObslugiwanieBledow1

1. Przeczytaj o metodzie int.Parse() (https://learn.microsoft.com/en-us/dotnet/api/system.int32.parse?view=net-7.0#system-int32-parse(system-string)) i zobacz jakie błędy może ona zwrócić. Następnie napisz program który:
    a. Poprosi użytkownika o podanie liczby.
    b. Korzystając z metody int.Parse przekonwertuje podany string na liczbę typu int.
    c. Na koniec program ma w przystępny dla użytkownika sposób wyświetlić przekonwertowaną liczbę.
    d. Program ma obsłużyć wszystkie błędy jakie może zwrócić ta metoda i wyświetlić odpowiednie (inne od siebie) komunikaty dla użytkownika informujące co źle zrobił.
    e. Przygotuj kilka przykładów które wywołają każde z tych błędów w celu przetestowania programu (tzn. że należy przeczytać JAK wywoływane są te błędy i przygotować przykłady, które te błędy wywołają).
    f. Opcjonalnie – program ma cały czas prosić o podanie danych i wyświetlać wynik aż użytkownik nie napisze tekstu „exit”.

2. Przeczytaj o metodzie string.CopyTo() (https://learn.microsoft.com/en-us/dotnet/api/system.string.copyto?view=net-7.0) i zobacz jakie błędy może ona zwrócić. Następnie napisz program który:
    a. Poprosi użytkownika o podanie zdania.
    b. Następnie poprosi o podanie liczby która będzie informować od jakiego indeksu ma się zacząć wycinanie tekstu.
    c. Następnie poprosi o podanie ilości znaków, jakie mają zostać wycięte z tekstu.
    d. Po podaniu danych przez użytkownika (i sprawdzeniu, czy są poprawne) korzystając z metody string.CopyTo() wyciągnij z przekazanego zdania ten ciąg znaków, który wskazał użytkownik (czyli użytkownik podaje najpierw zdanie będące sourcem, potem

indeks od którego ma się zacząć wycinanie i na koniec ilość znaków do wycięcia z tego pierwotnego tekstu).

e. Na koniec program ma w przystępny dla użytkownika sposób wyświetlić wynik wycięcia tekstu z przekazanego zdania.
f. Program ma obsłużyć wszystkie błędy jakie może zwrócić ta metoda i wyświetlić odpowiednie (inne od siebie) komunikaty dla użytkownika informujące co źle zrobił.
g. Przygotuj kilka przykładów które wywołają każde z tych błędów w celu przetestowania programu (tzn. że należy przeczytać JAK wywoływane są te błędy i przygotować przykłady, które te błędy wywołają).
h. Opcjonalnie – program ma cały czas prosić o podanie danych i wyświetlać wynik aż użytkownik nie napisze tekstu „exit”.

Przykład efektów działania w konsoli:

Podaj zdanie:

Tomek ulozyl nam strasznie dlugie i meczace zadania.

Podaj indeks od którego zacząć wycinanie znaków:

4

Podaj ilość znaków do wycięcia:

20

Wycięto tekst:

k ulozyl nam straszn

3. Przeczytaj o metodzie decimal.Divide (https://learn.microsoft.com/en-us/dotnet/api/system.decimal.divide?view=net-7.0) i zobacz jakie błędy może ona zwrócić. Następnie napisz program który:
    a. Poprosi użytkownika o podanie pierwszej liczby.
    b. Następnie poprosi użytkownika o podanie drugiej liczby.

c. Korzystając z metody decimal.Parse przekonwertuje podane liczby na typ decimal (UWAGA: mają zostać obsłużone wszystkie błędy które metoda decimal.Parse może zwrócić – należy przeczytać w dokumentacji jakie to mogą być Exceptions).
d. Na koniec program ma w przystępny dla użytkownika sposób wyświetlić wynik dzielenia podanych liczb.
e. Program ma obsłużyć wszystkie błędy jakie może zwrócić ta metoda i wyświetlić odpowiednie (inne od siebie) komunikaty dla użytkownika informujące co źle zrobił. (CZYLI: ma obsłużyć błędy i metody decimal.Parse i metody decimal.Divide).
f. Przygotuj kilka przykładów które wywołają każde z tych błędów w celu przetestowania programu (tzn. że należy przeczytać JAK wywoływane są te błędy i przygotować przykłady, które te błędy wywołają).
g. Opcjonalnie – program ma cały czas prosić o podanie danych i wyświetlać wynik aż użytkownik nie napisze tekstu „exit”.