# Házi Feladat

A feladat megoldását a Program.cs fájlba készítse el, melyet beadás előtt nevezzen át. A beadandó forrásfájl elnevezése a feladat azonosítója és a saját neptunkódja legyen alulvonással elválasztva, nagybetűkkel: **AZONOSÍTÓ_NEPTUNKOD.cs**

A feladattal kapcsolatos további információk az utolsó oldalon találhatók (ezen ismeretek hiányából adódó reklamációt nem fogadunk el!).

---

Valósítsa meg a Caesar kódolás általánosított változatát (ROT-X), ami a bemenetként megadott üzenetet ($S$) az eltolás mértékének ($X$) megfelelően alakítja át és jeleníti meg a kimeneten.

A Caesar kódolás egy egyszerű eltolását jelenti az üzenet karaktereinek az ábécében ($D$) 13-mal (vagyis a ROT-13 a Caesar kódolás), aminek általánosítása, mikor az eltolás mértékét szabadon választják 0 és 36 között (számokkal kiegészített angol ábécé esetén).

A feladatmegoldás során használjuk az angol ábécét számokkal kiegészítve, illetve az angol üzeneteket. Továbbá figyeljünk a kis- és nagybetűkre, illetve, hogy az írásjeleket az algoritmus ne változtassa meg (lásd: példa).

### *Bemenet (Console)*

- első érték: $S$ - az elkódolandó üzenet
- második érték: $X$ - az egyes karakterek eltolásának mértéke

### *Kimenet (Console)*

- az elkódolt üzenet

### *Megkötések*

- $0 \leq X \leq 36$
- $0 \leq S\ karaktereinek\ száma \leq 1\ 000$
- minden karakter az $S$-ben, $S[i] \in \{0-9, a-z, A-Z, -.?!, szóköz, vessző\}$
- $D \in \{ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789\}$
- az ábécében nem szereplő karaktereket a kódolás nem változtatja meg
- a karakterek írásmódját (kis-/nagybetű) - amennyiben ismert - a kódolás nem változtatja meg
- a `Main` metódus mellett egyetlen metódus legyen, mely első paramétere az elkódolandó üzenet, második az eltolás mértéke, visszatérési értéke pedig az elkódolt üzenet legyen

### *Példa*

Console input
```
1 Hello World!
2 13
```

Console output
```
1 Uryy1 914yq!
```

#### *Értelmezés*

A felhasználó a következő üzenetet szeretné elkódolni: `Hello World!`

A felhasználó a következő eltolást szeretné alkalmazni az üzeneten: `13`

Az eredeti ábécé: `ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789`

A 13-al eltolt ábécé: `NOPQRSTUVWXYZ0123456789ABCDEFGHIJKLM`

Az üzenet karaktereihez rendelt új érték:

```
H -> U          o -> 1          r -> 4
e -> r          'space' -> 'space'  l -> y
l -> y          W -> 9          d -> q
l -> y          o -> 1          !  -> !
```

Tehát a megjelenítendő elkódolt üzenet a következő: `Uryy1 914yq!`

***Tesztesetek***

Az alkalmazás helyes működését legalább az alábbi bemenetekkel tesztelje le!

| **A felhasználó által megadott bemenetek** | | **A bemenethez tartozó elvárt kimeneti értékek** |
|---|---|---|
| $S$ | $R$ | |
| `Hello World!` | 0 | `Hello World!` |
| `Hello World!` | 1 | `Ifmmp Xpsme!` |
| `Hello World!` | 13 | `Uryy1 914yq!` |
| `Hello World!` | 36 | `Hello World!` |
| `0123456789` | 33 | `XYZ0123456` |
| `?! - ,.` | 11 | `?! - ,.` |
| `Abcdef. GH-IJ! klmno 123456789!?!` | 26 | `012345. 67-89! abcde RSTUVWXYZ!?!` |
| `abcdefghijklmnopqrstuvwxyz` | 22 | `wxyz0123456789abcdefghijkl` |
| `ABCDEFGHIJKLMNOPQRSTUVWXYZ` | 22 | `WXYZ0123456789ABCDEFGHIJKL` |
| `0123456789` | 22 | `MNOPQRSTUV` |
| `Knowledge is Power` | 18 | `256e3wvyw 0a 76ew9` |
| `256e3wvyw 0a 76ew9` | 18 | `KNOwLedge Is POweR` |
| | 5 | |

*A táblázat utolsó példájában az S string üres, a kimenet ezesetben egy üres string!*

*A fenti tesztesetek nem feltétlenül tartalmazzák az összes lehetséges állapotát a be- és kimenet(ek)nek, így saját tesztekkel is próbálja ki az alkalmazás helyes működését!*

### *Tájékoztató*

A feladattal kapcsolatosan általános szabályok:

- A feladat megoldásaként beadni vagy a betömörített solution mappa egészét vagy a Program.cs forrásfájlt kell (hogy pontosan melyiket, azt minden feladat külön definiálja), melynek elnevezése a feladat azonosítója és a saját neptunkódja legyen alulvonással elválasztva, nagybetűkkel: **AZONOSÍTÓ_NEPTUNKOD**[.zip|.cs]
- A megvalósítás során lehetőség szerint alkalmazza az előadáson és a laboron ismertetett programozási tételeket és egyéb algoritmusokat.
- Az alkalmazás elkészítése során mindenesetben törekedjen a megfelelő típusok használatára, illetve az igényes (*formázott, felesleges változóktól, utasításoktól mentes*) kód kialakítására, mely magába foglalja az elnevezésekkel kapcsolatos ajánlások betartását is (bővebben).
- A megoldásokhoz nem használhatók a beépített rendezőmetódusok (például: `Array.Sort`), a LINQ technológia (`System.Linq`), kivételkezelés (`try-catch-finally` blokk), a `goto`, a `continue` és a `break` (kivéve a `switch-case` szerkezetnél) utasítások, az alábbi gyűjtemények: `ArrayList`, `List`, `SortedList`, `Dictionary`, `Stack`, `Queue`, `Hastable`, a `var` az `object` és a `dynamic` kulcsszavak, illetve figyelembe kell venni a *Megkötések* pontban meghatározott további szabályokat.
- A leadott feladat megoldással kapcsolatos minimális elvárás a leírásban feltüntetett tesztesetek helyes futtatása, a *Megkötések* pontban definiáltaknak való megfelelés, ezeket leszámítva viszont legyen kreatív a feladat megoldásával kapcsolatban.
- A kiértékelés során csak a *Megkötések* pont szerenti helyes bemenettel lesz tesztelve az alkalmazás, a "tartományokon" kívüli értéket nem kell lekezelnie az alkalmazásnak.
- **Ne másoljon vagy adja be más megoldását!** Minden ilyen esetben az összes (felépítésben) azonos megoldás duplikátumként lesz megjelölve, melyek közül kizárólag, az időrendben elsőnek leadott lesz elfogadva.
- **Idő után leadott vagy helytelen elnevezésű megoldás vagy a kiírásnak nem megfelelő megoldás vagy fordítási hibát tartalmazó vagy (helyes bemenetet megadva) futásidejű hibával leálló kód nem értékelhető!**
- A feladat leírása az alábbiak szerint épül fel (* - opcionális):
  - *Feladat leírása* - a feladat megfogalmazása
  - **Bemenet* - a bemenettel kapcsolatos információk
  - **Kimenet* - az elvárt kimenettel kapcsolatos információk
  - *Megkötések* - a bemenettel, a kimenettel és az algoritmussal kapcsolatos megkötések, melyek figyelembevétele és betartása kötelező, továbbá az itt megfogalmazott bemeneti korlátoknak a tesztek minden eseteben eleget tesznek, így olyan esetekre nem kell felkészülni, amik itt nincsenek definiálva
  - **Megjegyzések* - további, a feladattal, vagy a megvalósítással kapcsolatos megjegyzések
  - *Példa* - egy példa a feladat megértéséhez
  - *Tesztesetek* - további tesztesetek az algoritmus helyes működésének teszteléséhez, mely nem feltétlenül tartalmazza az összes lehetséges állapotát a be- és kimenet(ek)nek
- **Minden eseteben pontosan azt írja ki és olvassa be az alkalmazás, amit a feladat megkövetel, mivel a megoldás kiértékelése automatikusan történik!** Így például, ha az alkalmazás azzal indul, hogy kiírja a konzolra a *"Kérem az első számot:"* üzenetet, akkor a kiértékelés sikertelen lesz, a megoldás hibásnak lesz megjelölve, ugyanis egy számot kellett volna beolvasni a kiírás helyett.
- A kiértékelés automatikusan történik, így különösen fontos a megfelelő alkalmazás elkészítése, ugyanis amennyiben nem a leírtaknak megfelelően készül el a megoldás úgy kiértékelése sikertelen lesz, a megoldás pedig hibás.
- Az automatikus kiértékelés négy részből áll:
  - Unit Test-ek - az alkalmazás futásidejű működésének vizsgálatára
  - Szintaktikai ellenőrzés - az alkalmazás felépítésének vizsgálatára
  - Duplikációk keresése - az azonos megoldások kiszűrésére
  - Metrikák meghatározása - tájékoztató jelleggel
- A kiértékelésnek eredményéből egy HTML report generálódik, melyet minden hallgató megismerhet.