Úkol 4

Vytvořte program, který bude jednoduchým způsobem provádět asymetrický šifrovací algoritmus, založený na principu zavazadlového algoritmu. V souboru `private_key.txt` bude uveden soukromý klíč. Program bude očekávat tři argumenty: parametry *p* a *q* (např. *p*=31, *q*=105) a vstupní soubor, který se bude šifrovat. Program bude fungovat ve dvou fázích.

V první fázi program načte do paměti soukromý klíč a z něj vytvoří klíč veřejný dle předaných argumentů. Veřejný klíč uložíte do souboru `public_key.txt`. Při vytváření veřejného klíče ošetřete, zda jsou splněny všechny omezující podmínky na čísla *p* a *q*. Ověřte také, jestli je posloupnost soukromého klíče superrostoucí. V případě, že něco není splněno, vygenerujte chybové hlášení a program ukončete.

Druhou fází programu je šifrování (resp. dešifrování). Vstupní soubor (např. `input.txt`) si převeďte na bloky o takové velikosti, aby to bylo v souladu s algoritmem a velikostmi obou klíčů. Proveďte šifrování všech bloků a výsledek uložte do souboru `output.txt`.
Pokud by nastala situace, že by poslední blok byl neúplný zarovnejte ho zprava pomocí nul.

Pokud implementujete **Rozšířený Eukleidův algoritmus** pro hledání hodnoty p^{-1} (multiplikativní inverzní hodnoty parametru *p*) získáte 7 bodů. Pokud budete hledat hodnotu parametru p^{-1} pomocí „brute force cyklu“ bude možné získat pouze max 6 bodů.

V případě textového vstupu bude výstupem programu soubor `output.txt` se třemi řádky v následujícím formátu:
Hexadecimální reprezentace zašifrovaného vstupu
Hexadecimální reprezentace zpětně dešifrovaného vstupu
Textová čitelná podoba zpětně dešifrovaného vstupu

Podobně jako u minulé úlohy dodržte prosím hexadecimální formát:
tedy např. `57 69 6b 69 70 65 64 ff` (včetně mezer po jednotlivých Bytech). Pokud šifrování a dešifrování proběhlo v pořádku, měl by se dešifrovaný text shodovat se souborem `input.txt`.

V případě obecného binárního souboru postupujte stejně jako u minulé úlohy a dešifrováním vytvořte opět původní soubor s předponou `knapsack_`, přičemž výsledný soubor bude nepoškozený a stejný jako soubor vstupní.

Hodnocení:

Max. počet bodů: **7**

Příklad spuštění:

```
java -jar knapsack_BIT.jar input.txt 31 105
//vytvořeny public_key.txt, output.txt

java -jar knapsack_BIT.jar dwarf_small.bmp 31 105
//vytvořeny public_key.txt, output.txt + knapsack_dwarf_small.bmp
```

V programu ošetřete všechny vstupy a případné výjimky.

Odevzdání:

Vytvořte a odevzdejte archiv `BIT_ukol4_<jméno_prijmeni>.zip` Součástí archivu budou zdrojové kódy programu a spustitelný soubor a také soubor `README.txt`. V něm stručně popište vaše řešení a případná omezení.

Řádný termín odevzdání: 20.4.2021 23:59:59
Mezní termín odevzdání: 27.4.2021 23:59:59