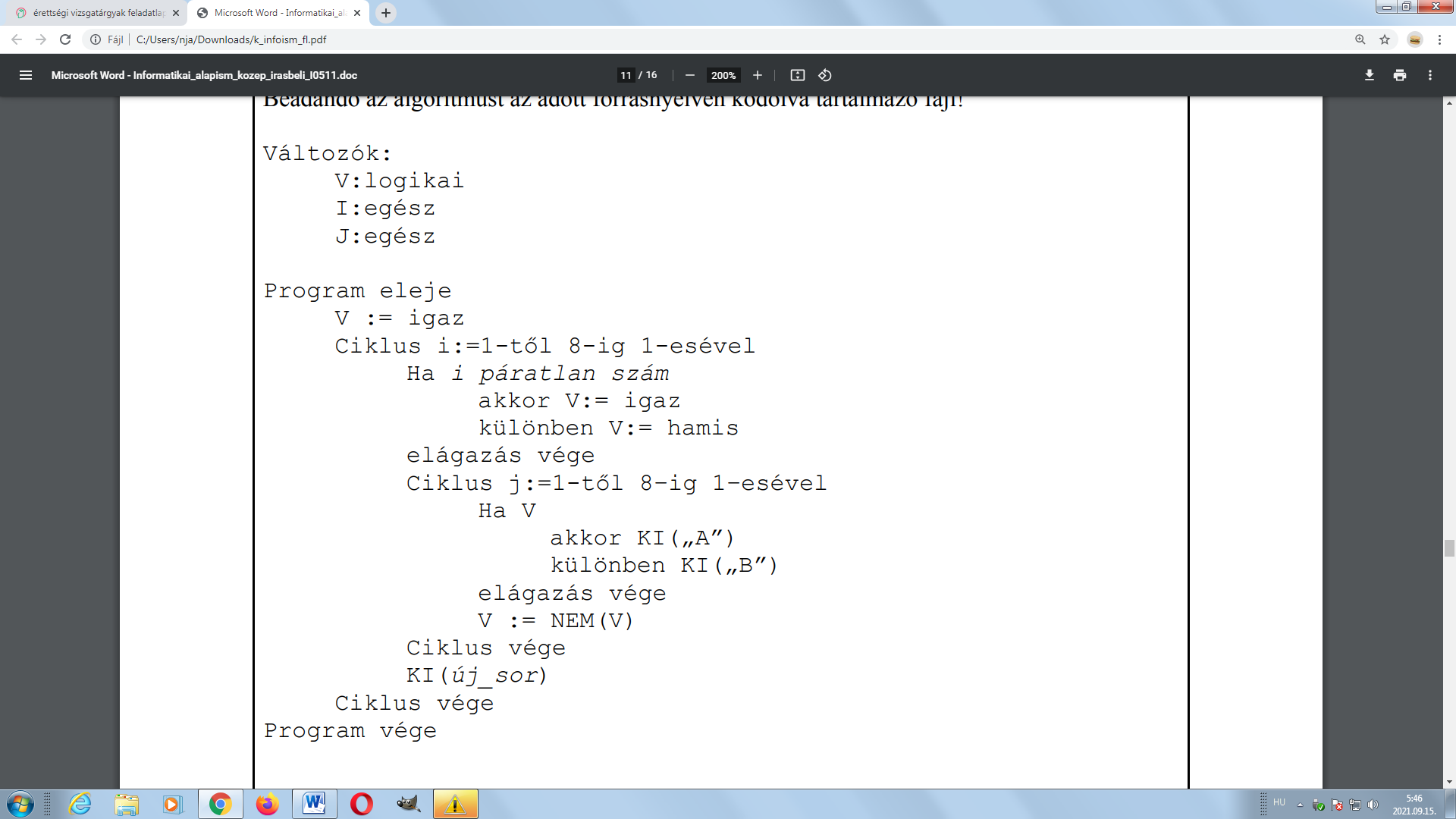
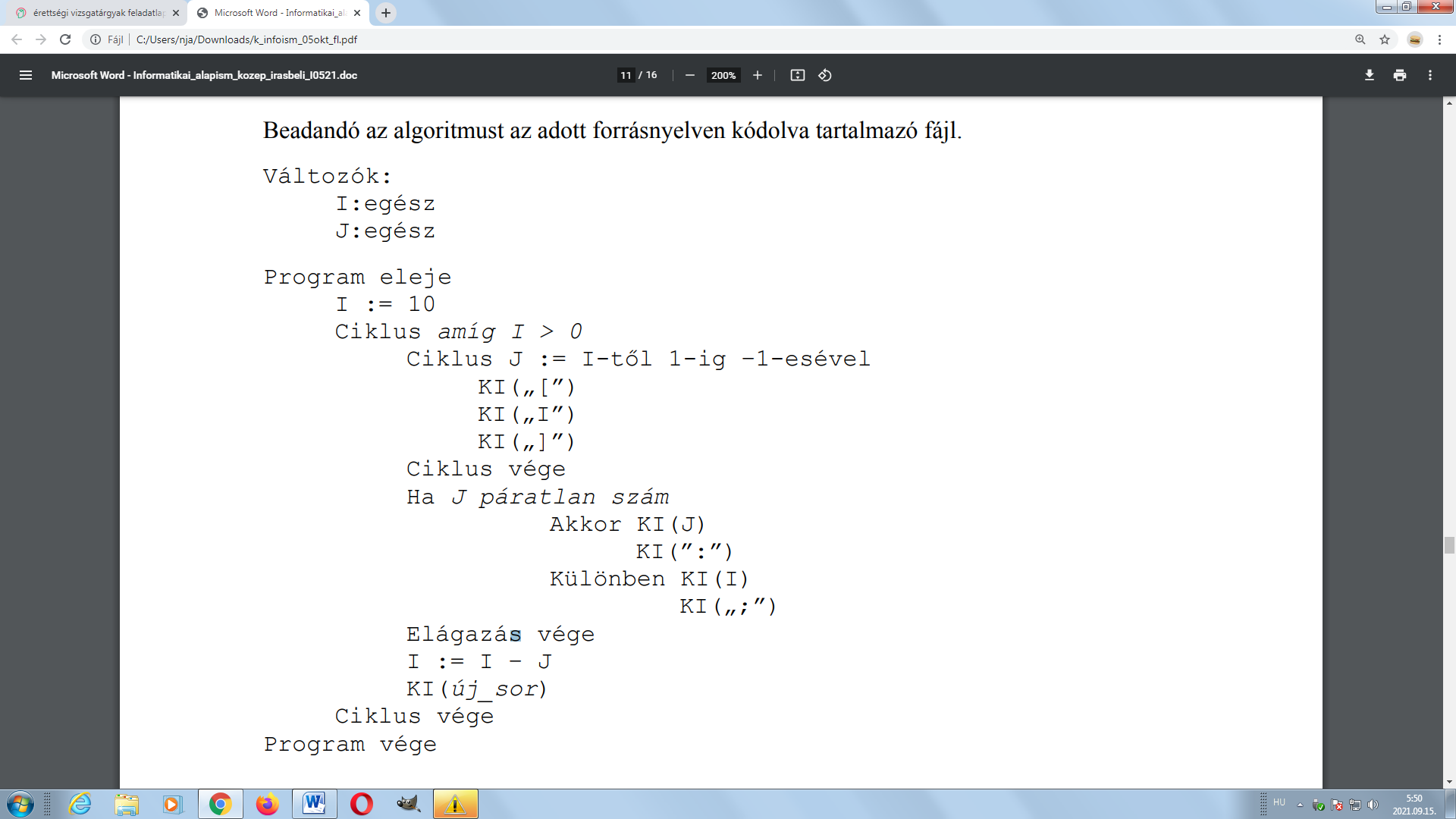
1. feladat

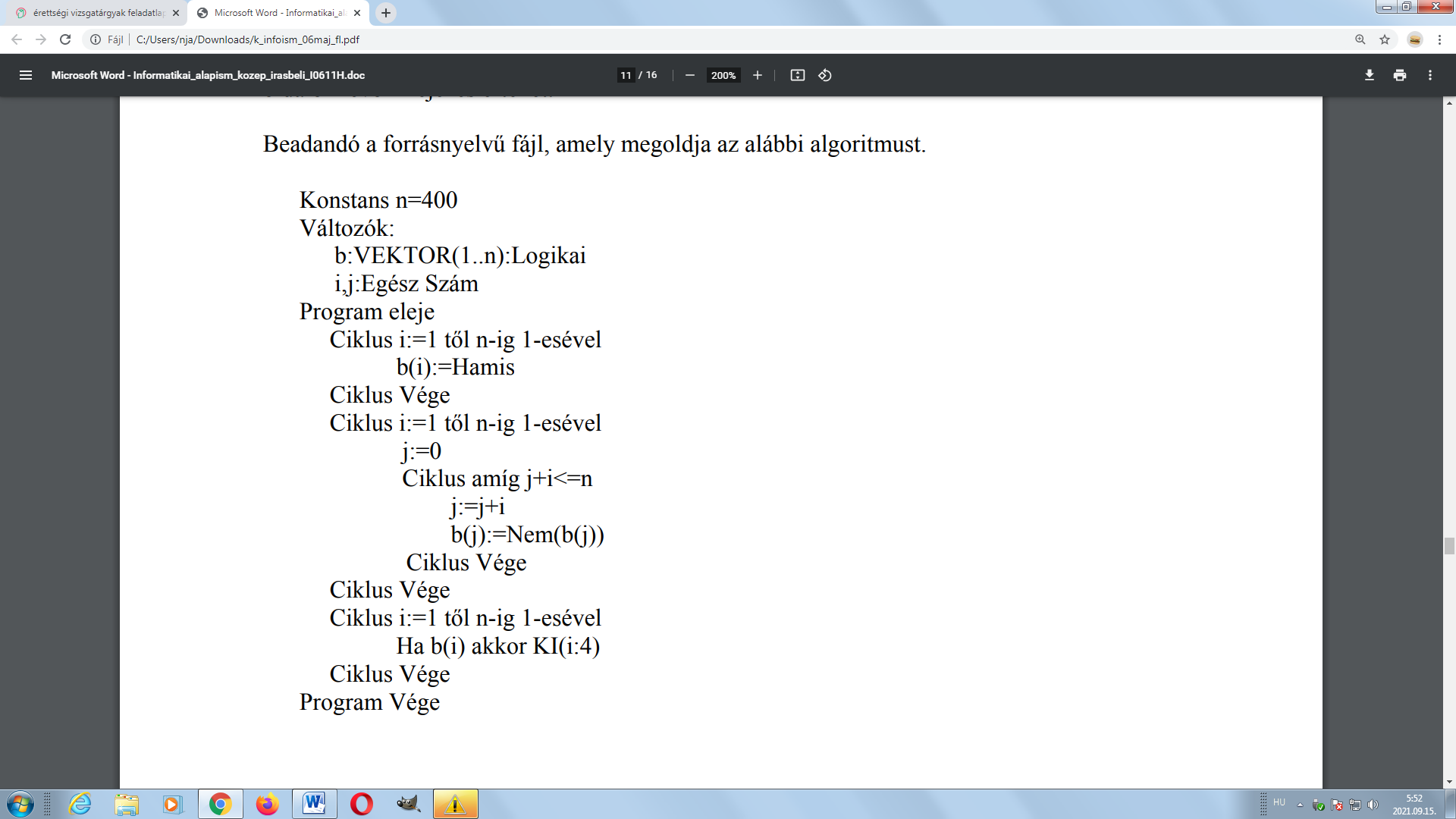
Kódolja az alábbiakban megadott algoritmust Pascal (C, JAVA, BASIC vagy egyéb által tanult) nyelven! A KI(X) eljárás az X szöveges kifejezés értékét írja a szabványos kimenetre (általában a képernyőre), az aktuális karakter-pozícióba. A KI(új\_sor) eljáráshívás egy „új sor” karaktert ír a kimenetre. A NEM(V) logikai függvény a paramétere tagadását adja eredményül. Amennyiben a használt programozási nyelv nem tartalmaz logikai típust, használjon helyette egész típusú változót, ahol a 0 reprezentálja a hamis, az 1 pedig az igaz értéket! A „:=” operátor az értékadást jelöli, mely az operátor bal oldalán megadott helyen (a változóban) tárolja a jobb oldalán megadott kifejezés értékét. Beadandó az algoritmust az adott forrásnyelven kódolva tartalmazó fájl!



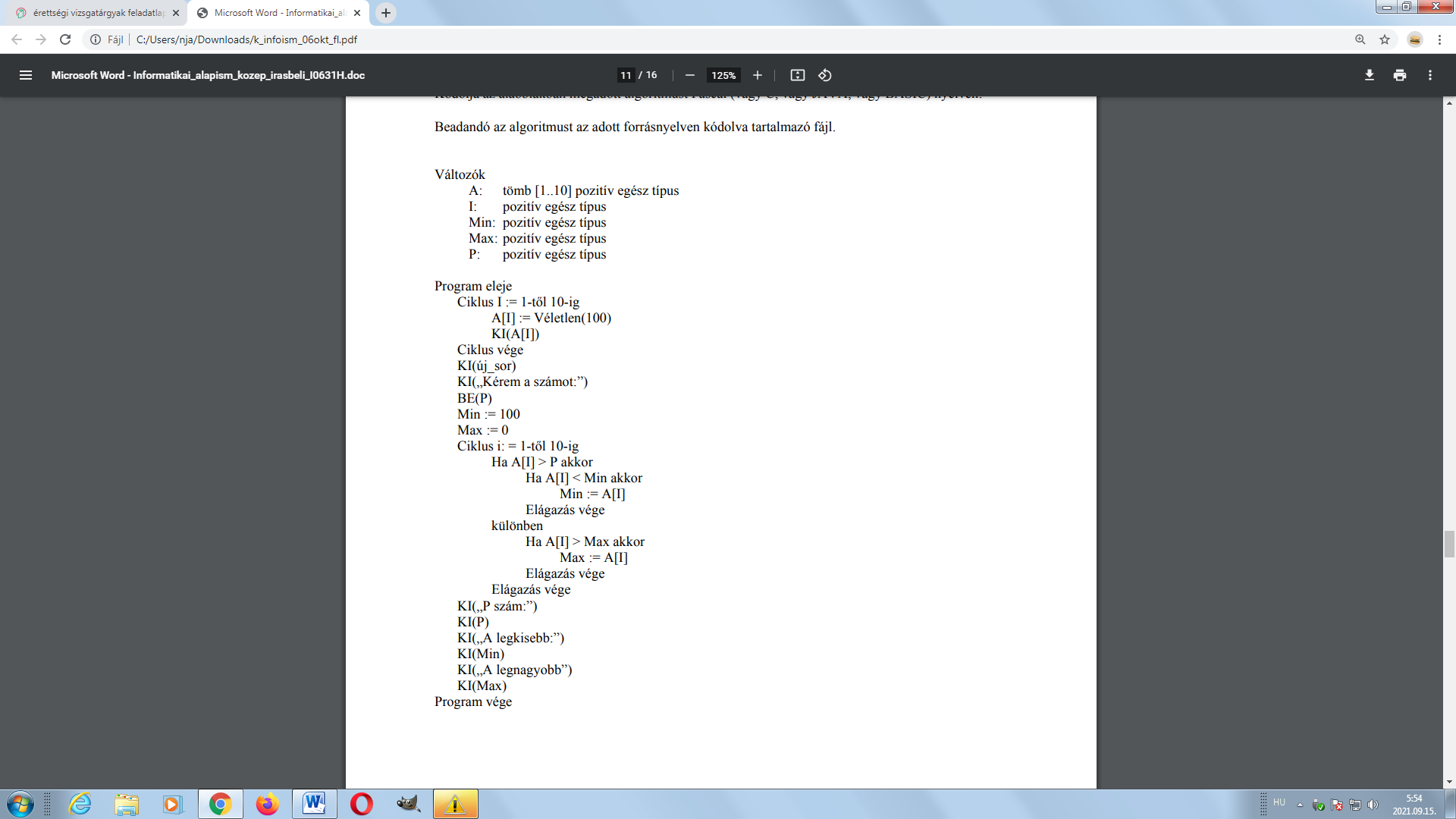
1. Kódolja az alábbiakban megadott algoritmust Pascal (vagy C, vagy JAVA, vagy BASIC) nyelven! A KI(X) eljárás az X szöveges kifejezés értékét írja a szabványos kimenetre (általában a képernyőre), az aktuális karakterpozícióba! A KI(új\_sor) eljáráshívás egy „új sor” karaktert ír a kimenetre. A NEM(V) logikai függvény a paramétere tagadását adja eredményül. Amennyiben az ön által használt programozási nyelv nem tartalmaz logikai típust, használjon helyette egész típusú változót, ahol 0 reprezentálja a hamis, az 1 pedig az igaz értéket! A „:=” operátor az értékadást jelöli mely az operátor bal oldalán megadott helyen (változóban) tárolja a jobb oldalán megadott kifejezés értékét! Beadandó az algoritmust az adott forrásnyelven kódolva tartalmazó fájl.



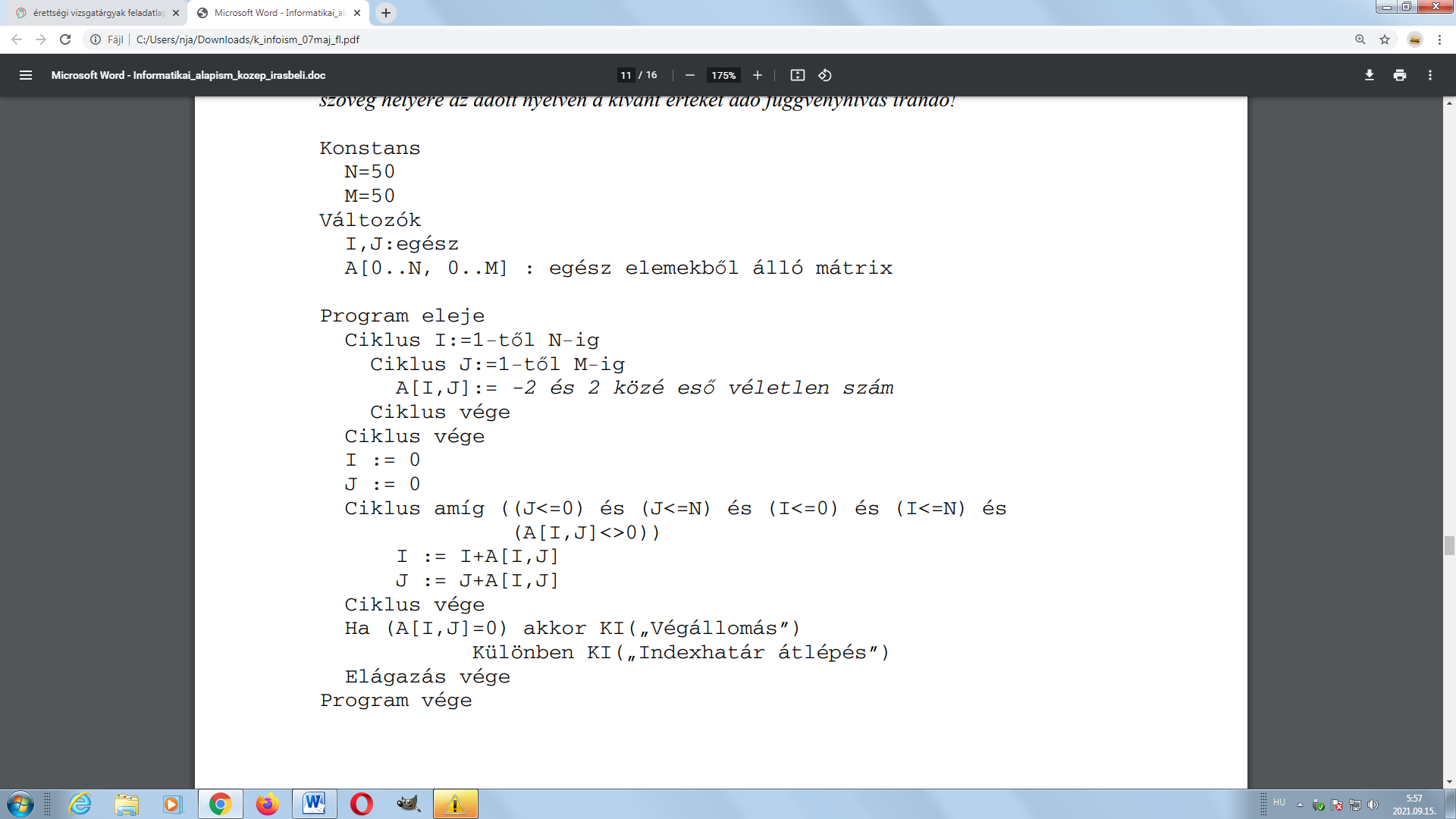
1. Tömbös feladat



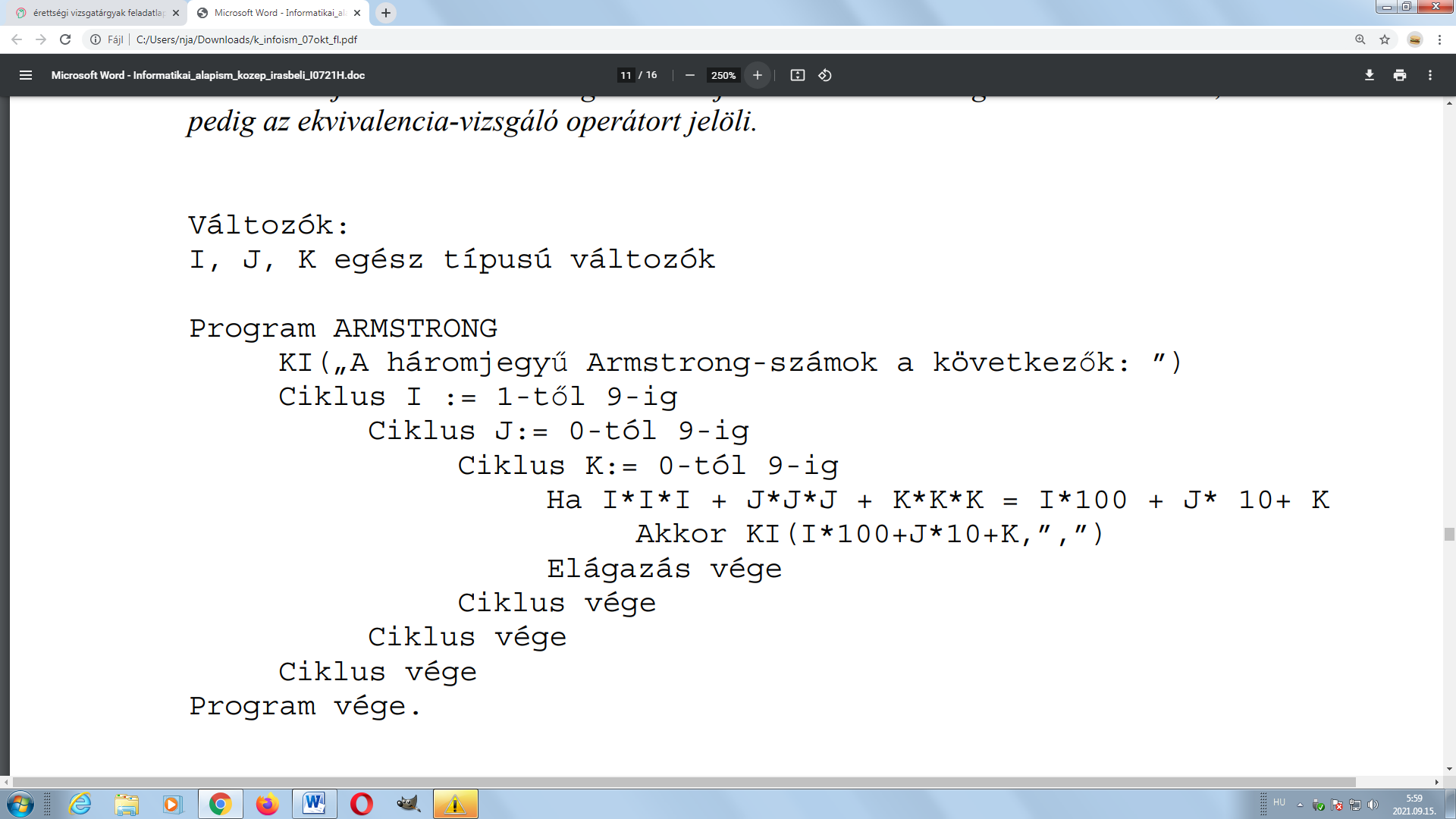
1. Legkisebb-legnagyobb



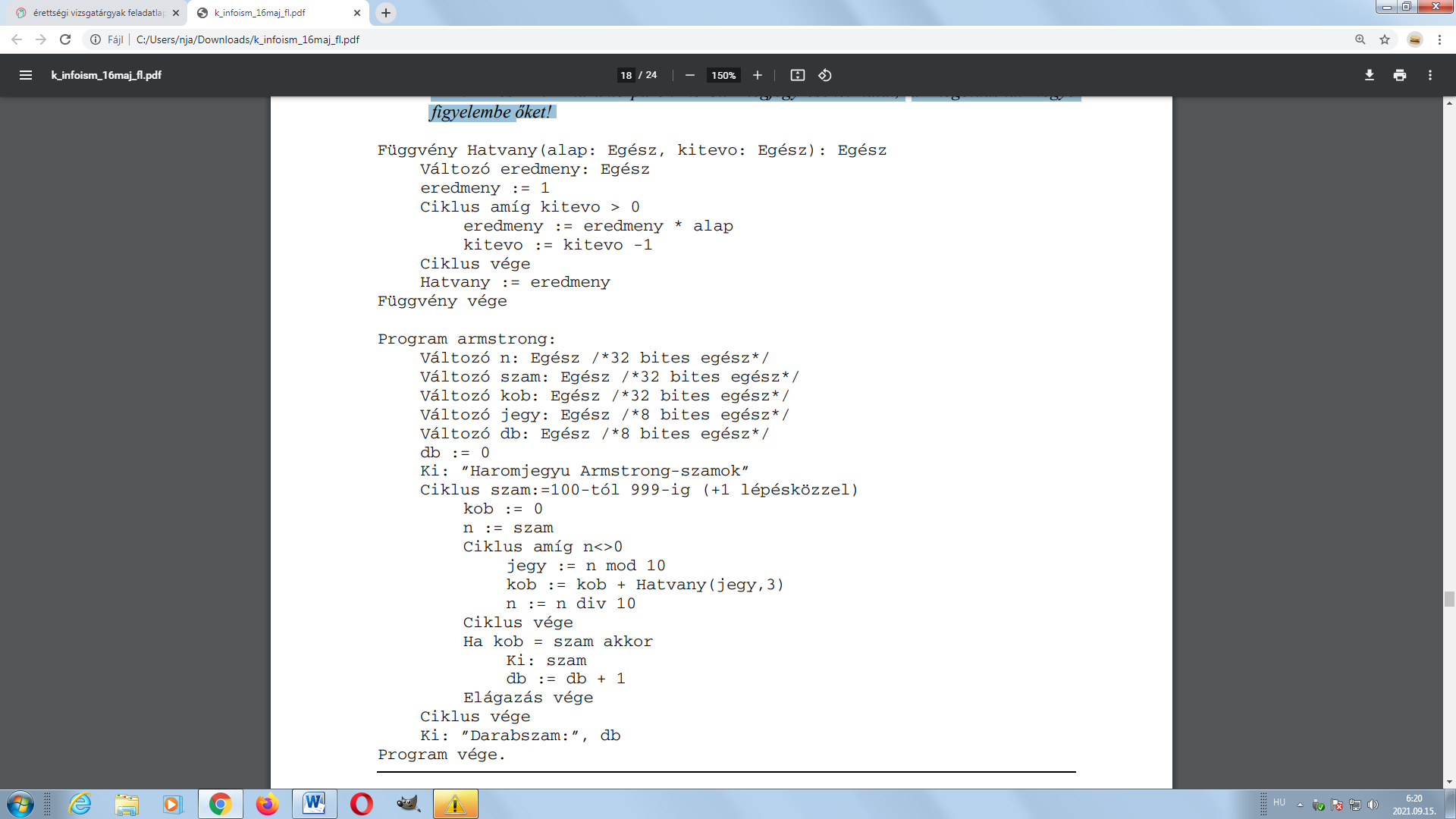
1. Kódolja az alábbi algoritmust az alábbi programozási nyelvek egyikén: Pascal, Java, BASIC, C, C++ vagy C# ! A feladat megoldásaként teljes, fordítható és futtatható kódot kérünk, mely az adatokat billentyűzetről (standard input) olvassa, és a képernyőre (standard output) írja ki. Vizuális fejlesztőeszköz használata esetén az algoritmust konzol alkalmazásként (szöveges ablakban futó) kérjük elkészíteni! A feladatban szereplő KI( ) eljárás paramétereit a sztenderd kimenetre írja! A konstans szövegeket idézőjelek között adtuk meg. A „:=” az értékadó, az „=” pedig az ekvivalencia vizsgáló operátort jelöli. A „–2 és 2 közé eső véletlen szám” szöveg helyére az adott nyelven a kívánt értéket adó függvényhívás írandó!



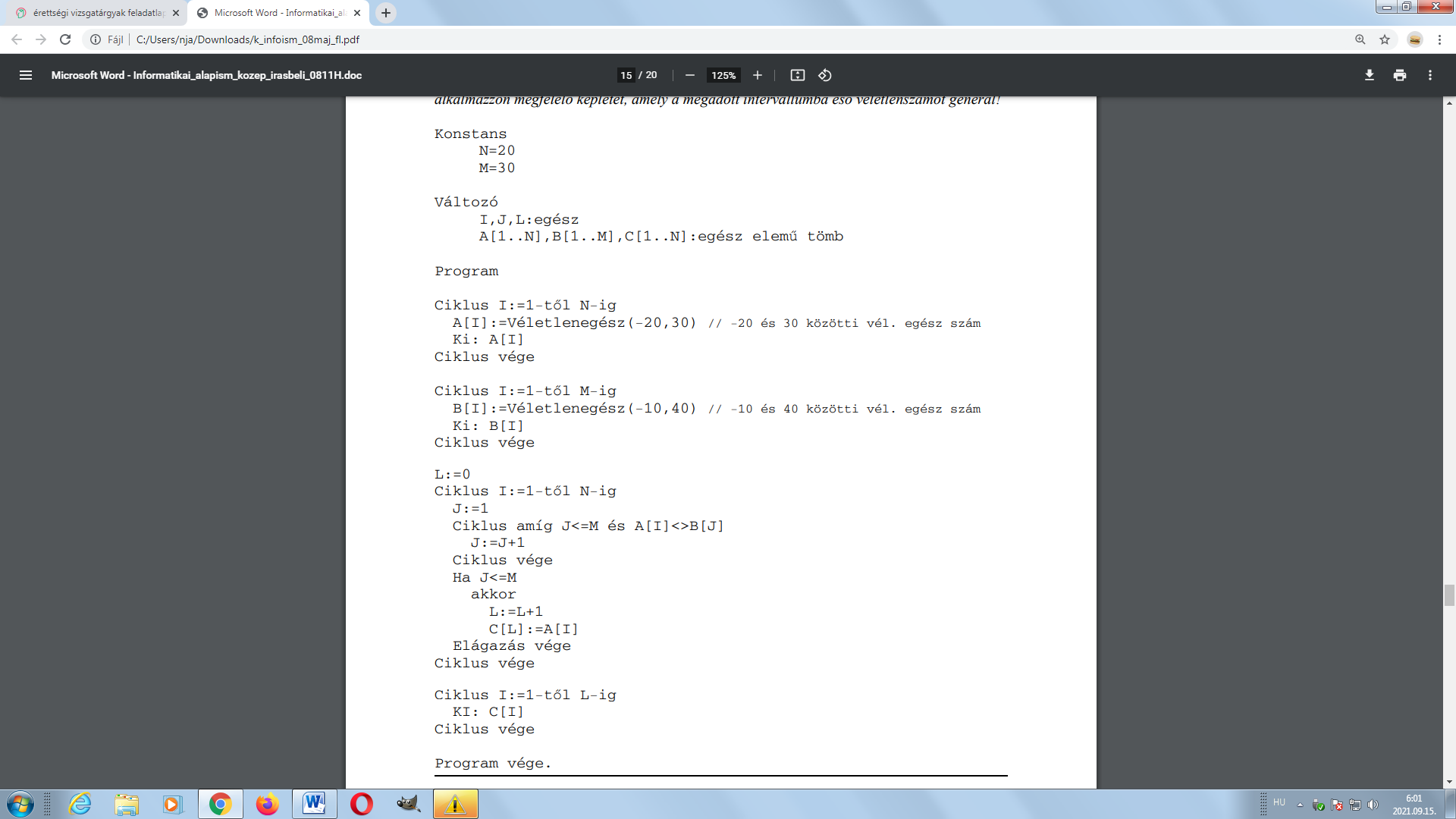
1. Amstrong számok A következő algoritmus a háromjegyű (N=3) Armstrong-számokat keresi meg és írja a képernyőre. Kiírásra kerül még az ilyen számok darabszáma is. (Armstrong-számnak nevezünk egy N jegyű számot, ha minden számjegyét az N-edik hatványra emeljük, majd a hatványokat összeadva az eredeti számot kapjuk.) Kódolja az alábbi algoritmust a választott programozási nyelven! Az elkészült program forráskódját mentse armstrong néven! A megoldás során vegye figyelembe a következőket: • A „div” az egészosztás, a „mod” a maradékképzés operátora. • A választott programozási nyelvtől függően eltérő jelölésű operátorokat és függvényeket kell alkalmaznia. • A ”/\*” és ”\*/” karakterpárok között megjegyzéseket talál, a megoldásnál vegye figyelembe őket!



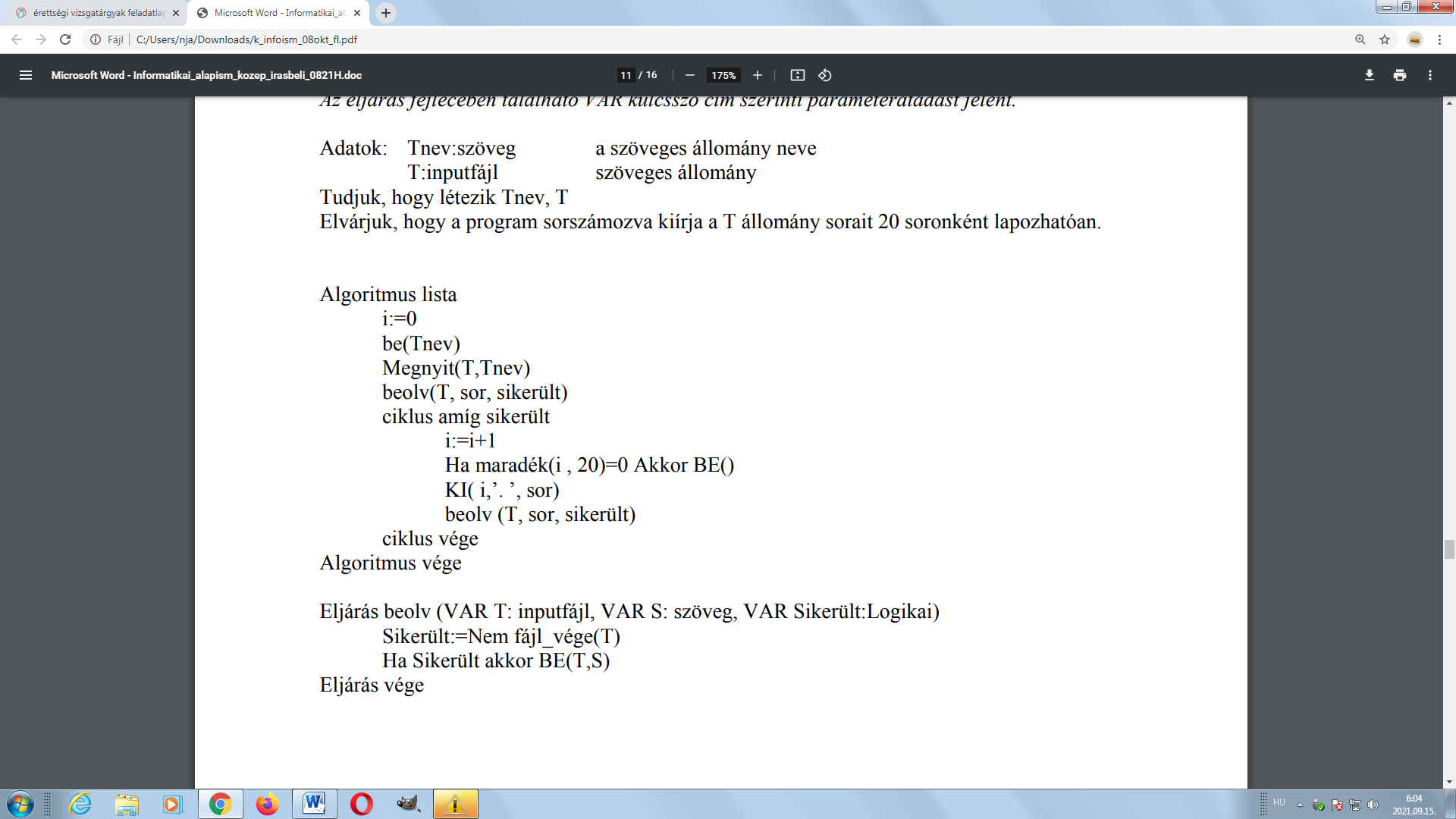
Másik megoldás



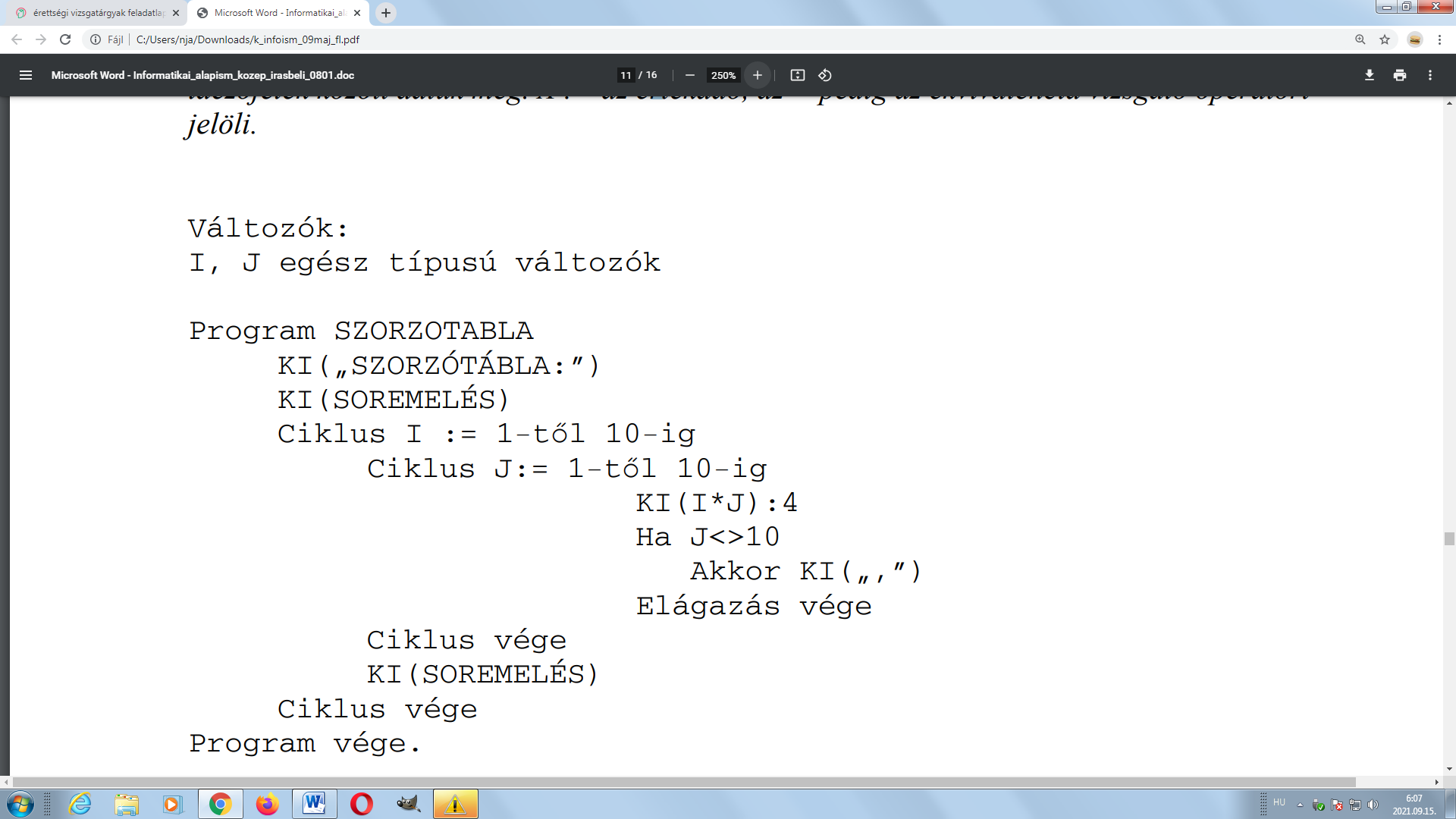
1. Mit csinál?



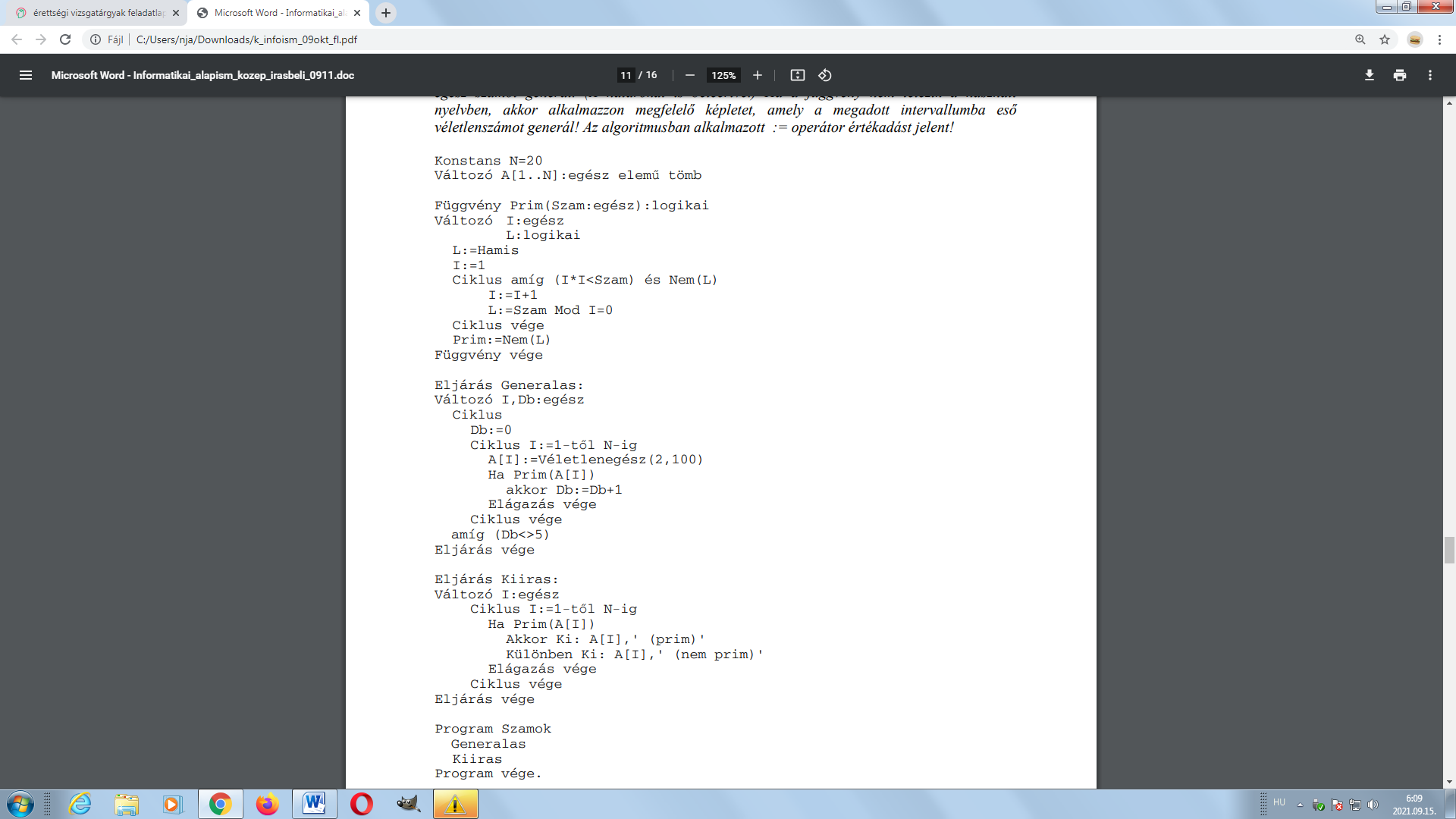
1. Fájlkezelés! Kódolja az alábbi algoritmust az alábbi programozási nyelvek egyikén: Pascal, Java, BASIC, C, C++ vagy C# ! A feladat megoldásaként teljes, fordítható és futtatható kódot kérünk, mely az adatokat billentyűzetről (standard input) olvassa, és a képernyőre (standard output) írja ki. A feladatban szereplő KI(…) eljárás paramétereit a standard kimenetre írja! A BE(…) eljárás a standard bemenetről olvas a paraméterben megadott változóba, ha van ilyen, kivéve, ha az első paraméter egy inputfájl. Utóbbi esetben az olvasást a fájlból kíséreli meg. Amennyiben a BE ( ) eljárásnak nincs paramétere, akkor az „>>ENTER<



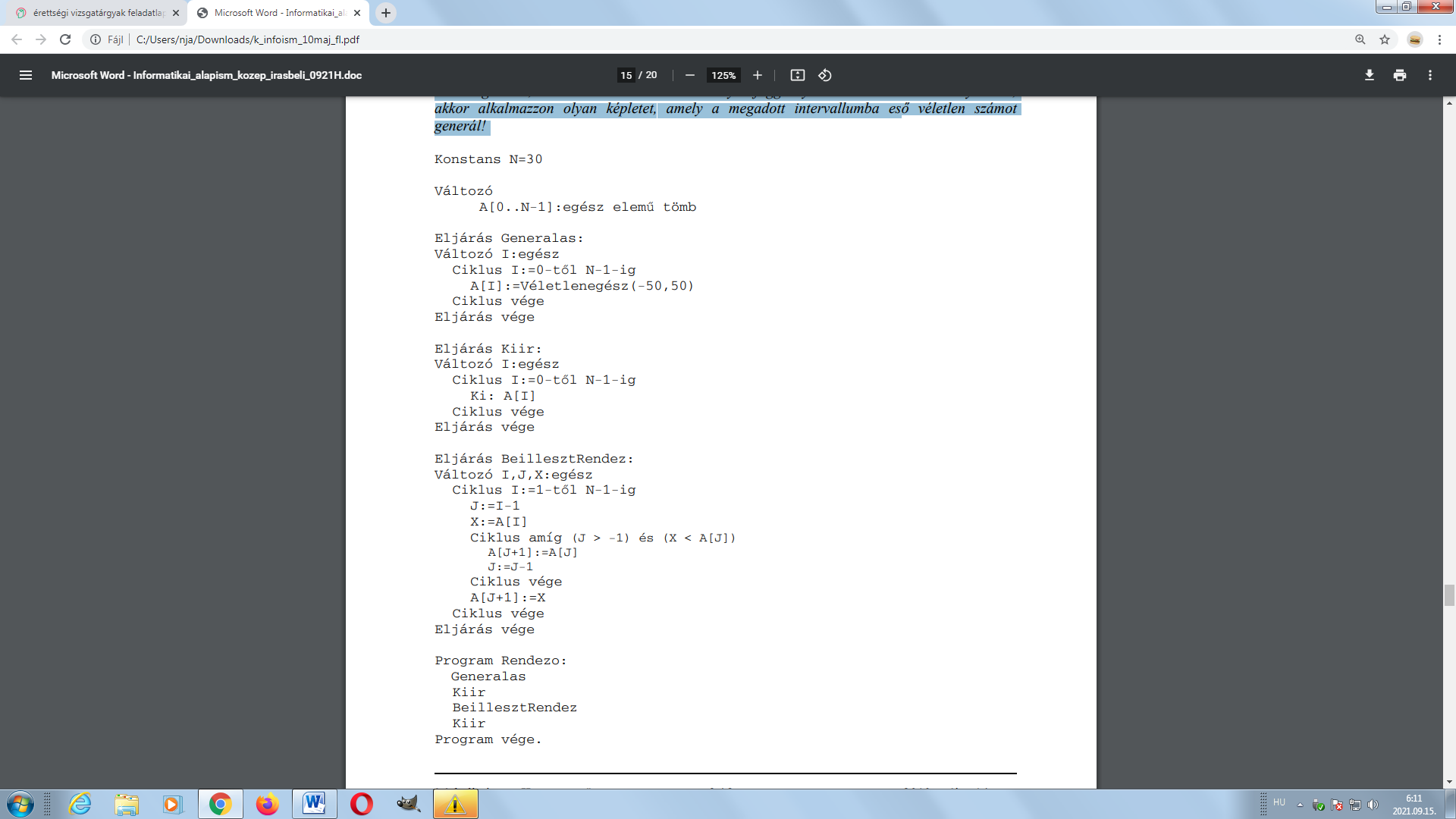
1. Szorzótábla! Kódolja az alábbi algoritmust az aktuális szoftverjegyzékben előzetesen megadott programozási nyelvek egyikén! A feladat megoldásaként teljes, fordítható és futtatható kódot kérünk, mely az adatokat billentyűzetről (standard input) olvassa, és a képernyőre (standard output) írja ki. Vizuális fejlesztőeszköz használata esetén az algoritmust konzol alkalmazásként (szöveges ablakban futó) kérjük elkészíteni! A feladatban szereplő KI( ) eljárás paramétereit a standard kimenetre írja! A KI(SOREMELÉS) eljáráshívás egy következő sor elejére állítja a kurzort. A KI():n hívás hatására a KI eljárás paraméterét egy n hosszúságú területre írja ki, amennyiben a kifejezés n-nél rövidebb, a fennmaradó helyeket szóközzel tölti fel. A konstans szövegeket idézőjelek között adtuk meg. A := az értékadó, az = pedig az ekvivalencia vizsgáló operátort jelöli.



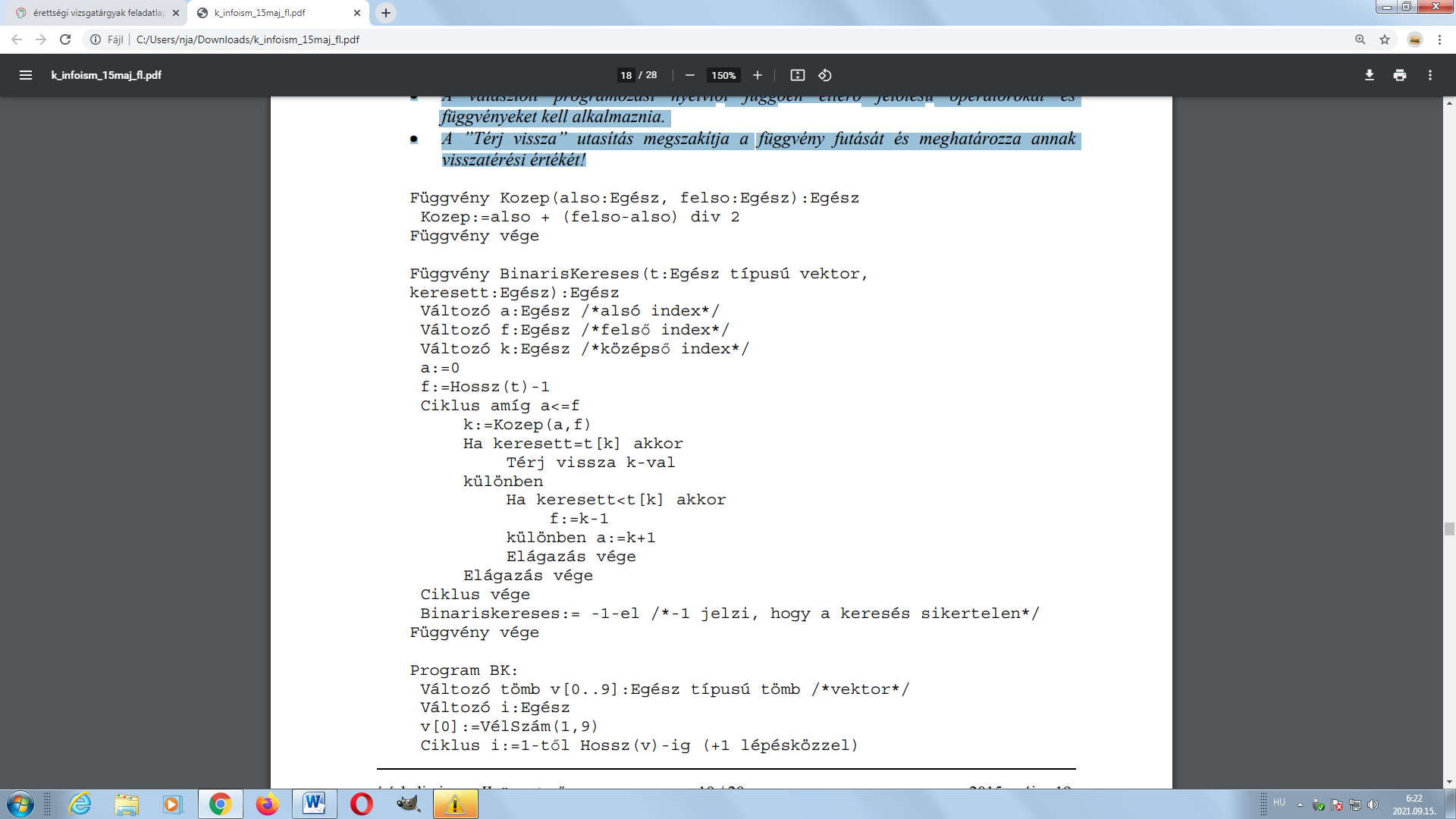
1. Prim! Az alábbi algoritmus olyan 20 elemű, véletlen egész számokból álló sorozatot állít elő, amelyben pontosan 5 db prím van. Kódolja az algoritmust az alábbi programozási nyelvek egyikén: Pascal, Java, BASIC, C, C++ vagy C# ! A feladat megoldásaként teljes, fordítható és futtatható kódot kérünk, mely az adatokat – ha szükséges – a billentyűzetről (standard input) olvassa, és a képernyőre (standard output) írja ki. Vizuális fejlesztőeszköz használata esetén az algoritmust konzol alkalmazásként (szöveges ablakban futó) kérjük elkészíteni! A Véletlenegész függvény a megadott határok közötti véletlen egész számot generál. (A határokat is beleértve.) Ha a függvény nem létezik a használt nyelvben, akkor alkalmazzon megfelelő képletet, amely a megadott intervallumba eső véletlenszámot generál! Az algoritmusban alkalmazott := operátor értékadást jelent!



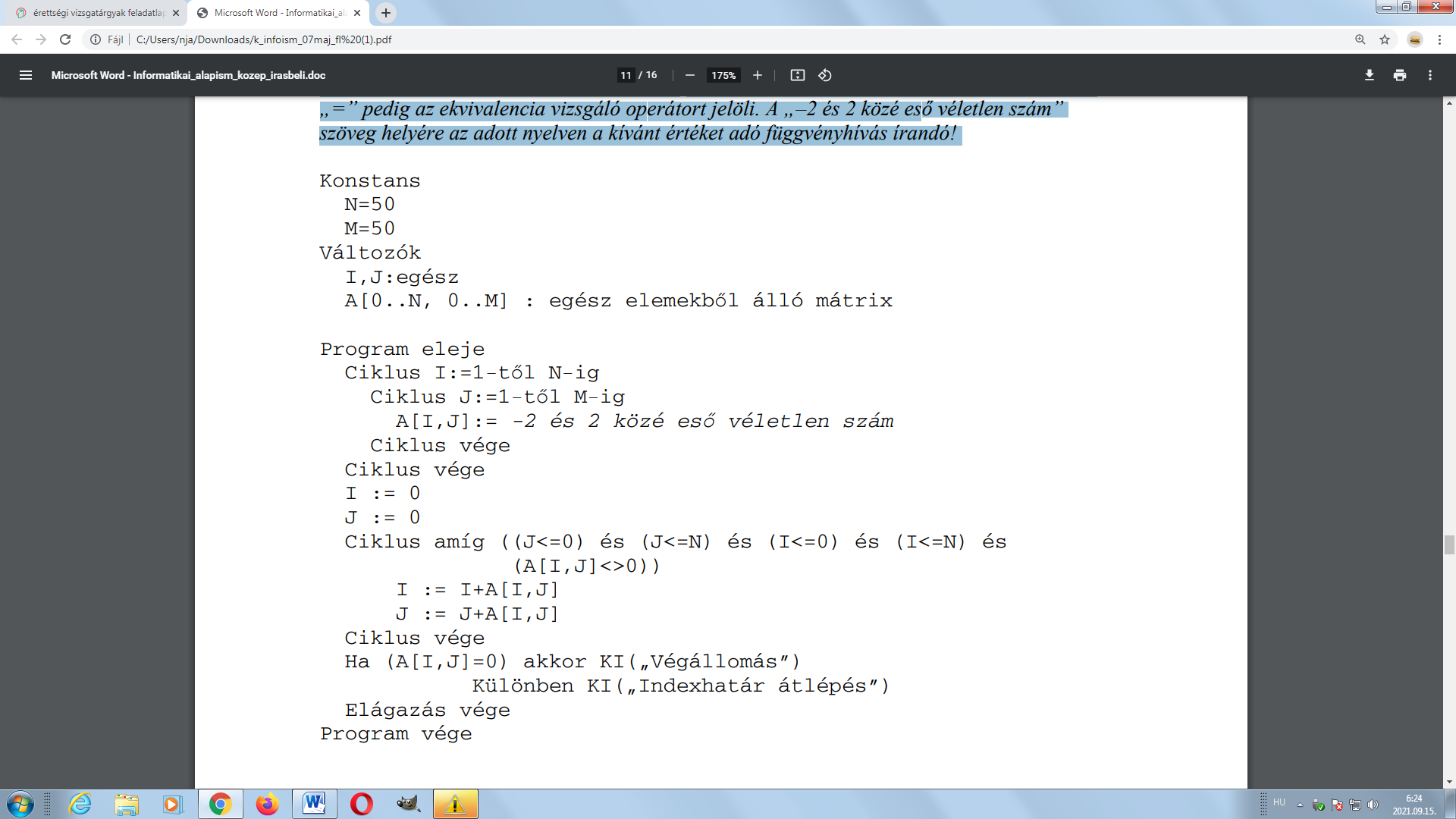
1. Rendezés! Kódolja az alábbi algoritmust! Beadandó a feladatot megoldó program forráskódja! A feladat megoldásaként teljes, fordítható és futtatható kódot kérünk, mely az adatokat a képernyőre (standard output) írja ki. Vizuális fejlesztőeszköz használata esetén a megoldást konzol (szöveges ablakban futó) alkalmazásként kérjük elkészíteni! Az algoritmusban használt Véletlenegész függvény a megadott határok közötti véletlen egész számot generál, a határokat is beleértve. Ha ilyen függvény nem létezik a használt nyelvben, akkor alkalmazzon olyan képletet, amely a megadott intervallumba eső véletlen számot generál!



1. Bináris keresés! A következő algoritmus egy véletlen számokkal növekvő rendben feltöltött vektorban keres egy számot a bináris keresés algoritmusával. Az algoritmus a keresett számot is véletlenszerűen állítja elő. A keresés eredményét a képernyőn jeleníti meg. Kódolja az algoritmust a választott programozási nyelven! A ”/\*” és ”\*/” karakterpárok között megjegyzéseket talál, ezeket helyezze el a megoldásban is! Az elkészült program forráskódját mentse BK néven! A megoldás során vegye figyelembe a következőket: • A ”div” az egészosztás operátora. • A ”Hossz()” függvény a vektor elemszámát adja meg. • A ”VélSzám(min, max) függvény véletlen számot generál (min<=VélSzám<=max) • A választott programozási nyelvtől függően eltérő jelölésű operátorokat és függvényeket kell alkalmaznia. • A ”Térj vissza” utasítás megszakítja a függvény futását és meghatározza annak visszatérési értékét!



1. Véletlen szám! Kódolja az alábbi algoritmust az alábbi programozási nyelvek egyikén: Pascal, Java, BASIC, C, C++ vagy C# ! A feladat megoldásaként teljes, fordítható és futtatható kódot kérünk, mely az adatokat billentyűzetről (standard input) olvassa, és a képernyőre (standard output) írja ki. Vizuális fejlesztőeszköz használata esetén az algoritmust konzol alkalmazásként (szöveges ablakban futó) kérjük elkészíteni! A feladatban szereplő KI( ) eljárás paramétereit a sztenderd kimenetre írja! A konstans szövegeket idézőjelek között adtuk meg. A „:=” az értékadó, az „=” pedig az ekvivalencia vizsgáló operátort jelöli. A „–2 és 2 közé eső véletlen szám” szöveg helyére az adott nyelven a kívánt értéket adó függvényhívás írandó!



1. Mit csinál?

