Készítsük el a `hd` (azaz hexdump) UNIX-utility egy egyszerűsített megvalósítását! A kapcsolókkal nem fogunk foglalkozni. A cél, hogy a szabványos bemenet, vagy megadott nevű fájlok tartalmát hexadecimális formában kiírjuk a szabványos kimenetre.

A megoldást lépésről lépésre készítjük el. Érdemes (pl. különböző könyvtárakban) az előző részfeladatok megoldását eltárolni. A programunk eredményét hasonlítgassuk össze a `hd` parancs eredményével, így tesztelhetjük a megoldásunkat!

Bemelegítés (1 pont)

Készítsünk programot, amely a fájl vége jelig olvas a szabványos bemenetről (használjuk a `getchar()` függvényt és az `EOF` értéket), és minden beolvasott karakter kódját hexadecimálisan kiírja! Minden kód kerüljön új sorba! Egy `unsigned char` kiírásához használjuk a `printf` függvény "'%02x" formázószabályát!

Futtatásnál irányítsuk pl. egy fájl tartalmát a szabványos bemenetre, vagy csak gépeljünk be valamit, és nyomjunk egy Ctrl-D-t az input végét jelzendő.

Példa kimenetre:

```
23
69
6e
63
6c
75
64
65
20
3c
```

Írjuk ki a hexadecimális kódokat 16-osával! (3 pont)

Ne írjuk ki azonnal minden beolvasott karakter hexadecimális kódját. Olvassunk be 16-ot egy megfelelő méretű tömbbe (a 16 értéket makróval adjuk meg a programban). Amikor a tömb megtelik, akkor írjuk ki a 16 karakter kódjait szóközökkel elválasztva, és egy soremeléssel lezárva. Emellett, amikor elfogynak a karakterek, az esetleg csak részben feltöltött tömbbe beolvasott karakterek kódjait is írjuk ki.

Vezessünk be ehhez egy `printline` nevű segédfüggvényt, mely megkapja paraméterként a kiírandó tömböt, valamint a tömbből kiírandó karakterek számát. (Általában 16-ot, a legutolsó hívásnál lehet, hogy kevesebbet.)

Egy sor kiírásánál megengedett, hogy vagy a legelső kiírt kód előtt, vagy a legutolsó kiírt kód után kiírjunk egy szóközt is!

Példa kimenetre:

```
09 69 66 28 20 30 20 3c 20 63 6f 75 6e 74 20 29
7b 0a 09 09 70 72 69 6e 74 6c 69 6e 65 28 6c 69
6e 65 2c 63 6f 75 6e 74 29 3b 0a 09 7d 0a 09 72
65 74 75 72 6e 20 30 3b 0a 7d 0a
```

A hexadecimális kódok mellett legyenek ott a karakterek is! (2 pont)

Egészítsük ki a sorok kiírását úgy, hogy miután a hexadecimális kódokat kiírtuk, a sor végére "|" jelekkel határolva írjuk ki a kapott (maximum 16) karaktert. Azokat a szóköztől különböző karaktereket, amelyekre a `ctype.h`-ban deklarált `isgraph()` függvény hamisat ad, a "." karakterrel helyettesítsük! Az egy sorban kiírt 16 számot középen plusz egy szóközzel tagoljuk!

Figyeljünk oda arra, hogy a 16 karakternél rövidebb sorok kiírásánál szóközökkel helyettesítsük a hiányzó kódokat, hogy a "|" karakterekkel határolt részek minden sorban egymás alá essenek!

Példa kimenetre:

```
23 69 6e 63 6c 75 64 65  20 3c 73 74 64 69 6f 2e  |#include <stdio.|
68 3e 0a 23 69 6e 63 6c  75 64 65 20 3c 63 74 79  |h>.#include <cty|
70 65 2e 68 3e 0a 0a                              |pe.h>..|
```

Írjunk ki minden sor elejére egy címet! (1 pont)

Minde sor elejére ki fogunk írni egy hexadecimális számot, mely azt jelzi, hogy az inputban hányadik karakter az, amellyel a sor kezdődik. Ehhez tartsuk nyilván egy `long` típusú változóban, hogy hol tartunk. Nulláról indítjuk, és minden teljes sor kiírása után 16-tal léptetjük. Adjuk át ezt is paraméterként a `printline` függvénynek, mely az értéket a `"%08lx"` formázóparanccsal írja ki.

Példa kimenetre:

```
00000000  23 69 6e 63 6c 75 64 65  20 3c 73 74 64 69 6f 2e  |#include <stdio.|
00000010  68 3e 0a 23 69 6e 63 6c  75 64 65 20 3c 63 74 79  |h>.#include <cty|
00000020  70 65 2e 68 3e 0a 0a                              |pe.h>..|
```

Használjuk a szabványos bemenetet fájlként! (1 pont)

Amit eddig a főprogram csinált, azt tegyük egy olyan függvénybe (pl. `hd` névvel), amely kap egy `FILE *` típusú paramétert. A `getchar()` függvény helyett a kapott fájlból olvassunk az `fgetc()` függvénnyel!

A főprogram hívja meg ezt a függvényt az `stdin` aktuális paraméterrel!

Olvassunk fájlból! (2 pont)

A főprogramot alakítsuk át úgy, hogy ha kap parancssori argumentumot, akkor az első parancssori argumentumot fájlnévként értelmezi, megnyitja olvasásra, és ennek a tartalmát dumpoljuk ki. Ha nem kap parancssori argumentumot, akkor – mint egészen idáig – a szabványos bemenetet dumpoljuk ki!

Ne felejtsük el bezárni a megnyitott fájlt! (`fopen`, `fclose`)

Ha a fájl megnyitása sikertelen, a programunk nem csinál semmit, de visszaad egy (nemnulla) hibakódot.

Dolgozzunk fel több input fájlt! (2 pont)

Bővítsük ki az előző megoldást úgy, hogy ne csak az első parancssori argumentumot használjuk, hanem az összeset! Egymás után mindegyik fájlt nyissuk meg, dumpoljuk ki a tartalmát és zárjuk be! A sorok elején szereplő címet minden fájl esetén kezdhetjük nullától!

Például, ha ugyanazt az `a.c` fájlt ki szeretném kétszer is dumpolni, akkor ez történik.

```
> ./hexdump a.c a.c
00000000  23 69 6e 63 6c 75 64 65  20 3c 73 74 64 69 6f 2e  |#include <stdio.|
00000010  68 3e 0a 23 69 6e 63 6c  75 64 65 20 3c 63 74 79  |h>.#include <cty|
00000020  70 65 2e 68 3e 0a 0a                              |pe.h>..|
00000000  23 69 6e 63 6c 75 64 65  20 3c 73 74 64 69 6f 2e  |#include <stdio.|
00000010  68 3e 0a 23 69 6e 63 6c  75 64 65 20 3c 63 74 79  |h>.#include <cty|
00000020  70 65 2e 68 3e 0a 0a                              |pe.h>..|
```

Olvasszuk egybe az inputokat! (3 pont)

A hivatalos `hd` program úgy működik, hogy az összes input fájlt egyben tekinti bemenetnek. Ez két dolgot is jelent: egyrészt nem kezdjük minden fájl dumpolásánál újra nullától a sor elejére kerülő címet, másrészt, ha egy fájl csonka sorra végződik, akkor azt kipótoljuk a következő fájlból. Alakítsuk át ilyenre a megoldásunk!

Arra is figyeljünk oda, hogy a `hd` parancs a legvégén kiírja még a címet egy üres sorba – ezt is írjuk bele a saját implementációnkba is!

Győződjünk meg arról (pl. `diff` segítségével), hogy az általunk adott kimenet megegyezik a `hd` program kimenetével!

```
> ./hexdump a.c a.c
00000000  23 69 6e 63 6c 75 64 65  20 3c 73 74 64 69 6f 2e  |#include <stdio.|
00000010  68 3e 0a 23 69 6e 63 6c  75 64 65 20 3c 63 74 79  |h>.#include <cty|
00000020  70 65 2e 68 3e 0a 0a 23  69 6e 63 6c 75 64 65 20  |pe.h>..#include |
00000030  3c 73 74 64 69 6f 2e 68  3e 0a 23 69 6e 63 6c 75  |<stdio.h>.#inclu|
00000040  64 65 20 3c 63 74 79 70  65 2e 68 3e 0a 0a        |de <ctype.h>..|
0000004d
```