A házik beadásához küldjétek el a megoldásokat (.c, .cpp, ha van .h és .hpp file-ok) csatolva a következõ emailcímre: tofihomework+2024info3hu@gmail.com

A lefordított programokat nem kell elküldeni. Ha úgy érzitek túl sok file-t küldenétek (5+) akkor bedobhatjátok egy zip-be, de nem muszáj.

LIFO

Implementáljuk egy LIFO (last-in-first-out) adatszerkezetet:

- Minden elem tároljon egy egész számot.
- Legyen **push** függvénye, ami egy új elemet helyez a lánc legelejére.
- Legyen **pop** függvénye, ami a lánc elsõ elemét törli és a benne tárolt értéket visszaadja.
- Írjunk hozzá **main** függvényt a teszteléshez!

Tipp a lap alján (ha van rá idõtök/kitartásotok, elõször próbáljátok meg nélküle megoldani/végiggondolni).

Az adatszerkezet stuktúrája jóformán megegyezik a gyakorlaton és elõadáson is használt láncolt listával. Az egyetlen különbség, hogy most nem a lista végére, hanem az elejére illesztjük az elemeket, de ugyanúgy next pointerrel mutatunk a következõ elemre.

Igazából a **push** egyszerûbb lesz, mint az append volt a láncolt listáknál, mert nem kell elnavigálnunk a lista végére sose. Mindig csak a **start** pointert kell átállítanunk az új elemre és az õ next-jét a korábbi **start** elemre. A **push** feje valahogy így nézhetne ki:

```
void push(struct lifo_e **start, int n);
```

A **pop** se bonyolult, hisz a **start** pointer tárolja a visszaadandó értéket szóval csak le kell mentenünk az értéket és a next pointerét, majd felszabadíthatjuk. Ne felejtsük el utána a **start** pointert átállítani a mentett next pointerre mielõtt visszaadjuk az értéket. A **pop** feje valahogy így nézhetne ki:

```
int pop(struct lifo_e **start);
```