# Välkmälu

**1 sek / 10 sek** **1 GB**

Mikrokontrolleri kiipi on sisse ehitatud $B$ bitine välkmälu. Meil on vaja seal hoida ja muuta $M$-bitist muutujat. Välkmälul on selline piirang, et bitte saab muuta `0`-st `1`-ks ükshaaval, kuid bitte `1`-st `0`-ks saab muuta ainult kogu mälu kustutades. Mälu saab kustutada ainult piiratud arv kordi, enne kui kiip kulub ära ja see tuleb välja vahetada. Sellepärast on soovitav olla võimeline kirjutama mällu võimalikult palju väärtuseid enne, kui kogu mälu kustutatakse.

Sinu ülesanne on välja mõelda efektiivne viis, kuidas salvestada väärtuseid välkmällu nii, et viimasena kirjutatud väärtus on alati loetav. Täpsemalt öeldes peab sinu programm realiseerima kaks operatsiooni:

- Väärtuse kirjutamine: Selle operatsiooni sisendiks on mälu algolek ja salvestatav väärtus. Väljastama peab mälu uue oleku.
- Väärtuse lugemine: Selle operatsiooni sisendiks on mälu olek pärast mingit hulka salvestamisi. Väljastama peab väärtuse, mis salvestati viimase kirjutamise operatsiooniga.

Sinu lugemise ja kirjutamise operatsioonid ei tohi omavahel mingil muul viisil informatsiooni vahetada, kui ainult lugeda sisendist mälu hetkeolekut ja kirjutamise operatsioonis muuta mõne (võimalusel mitte ühegi) biti väärtust `0`-st `1`-ks enne mälu uue oleku väljastamist.

**Suhtlus.** See on interaktiivne ülesanne. Kui su programm alustab, siis sisendi esimene rida sisaldab täisarvu $T$, kus $T = 0$ tähendab, et sinu programm kirjutab väärtuse mällu ja $T = 1$ tähendab, et see loeb väärtuse mälust. Teine rida sisaldab täisarve $B$ ja $M$. Järgnevad read kirjeldavad operatsioone.

Nii kirjutamise kui ka lugemise korral sisaldab iga operatsiooni esimene rida täisarvu $C$, kus $C = 0$ tähendab, et rohkem käsklusi ei tule ja su programm peaks töö lõpetama ning $C = 1$ tähendab, et programm peab jätkama.

- $T = 0$ ja $C = 1$ korral sisaldab teine rida kaht tühikuga eraldatud stringi: mälu algolekut $B$ bitise jadana ja salvestatavat uut väärtus $M$ bitise jadana. Kui su programm saab salvestada uue väärtuse mällu ainult mõnda bitti `0`-st `1`-ks muutes, siis peab see väljastama täisarvu 1 ja järgmisle reale mälu uue oleku $B$ bitise jadana. Kui su programm ei saa uut väärtust mällu salvestada, siis peab see väljastama täisarvu 0.
- $T = 1$ ja $C = 1$ korral sisaldab teine rida üht stringi: mälu olekut $B$ bitise jadana ja su programm peab väljastama ühe stringi: viimati mällu salvestatud väärtuse $M$ bitise jadana.

**Näide.**

| Sisend | Väljund |
|---|---|
| `0` | |
| `6 2` | |
| `1` | |
| `111111 00` | |
| | `0` |
| `1` | |
| `000000 11` | |
| | `1` |
| | `110000` |
| `0` | |

Selles näites on su programm pandud kirjutama 2-bitiseid väärtusi 6-bitisesse mällu. Esimeseks käskluseks on väärtuse `00` salvestamine, mida sinu programm ei ole võimeline täitma. Teiseks

käskluseks on väärtuse `11` kirjutamine, mida sinu programm saab täita. Pane tähele, et teise käskluse mälu algolek ei ole sama, mis sinu programmi väljund pärast esimest käsklust.

**Näide.**

| Sisend | Väljund |
|---|---|
| `1`<br>`6 2`<br>`1`<br>`110000` | |
| | `11` |
| `1`<br>`110100` | |
| | `01` |
| `0` | |

Selles näites on su programm pandud lugema 2-bitiseid väärtusi 6-bitisest mälust. Esimeseks käskluseks on väärtuse lugemine mäluolekust `110000`, kust su programm tagastab väärtuse `11`. Teiseks käskluseks on väärtuse lugemine mäluolekust `110100`, kust su programm tagastab väärtuse `01`.

**Märkus.** Selleks, et tagada sinu programmi vastuste jõudmine hindamiskeskkonda, pead sa iga väljastuse järel tühjendama väljundpuhvri (pane tähele, et ka viimane väljastatud rida lõpeb reavahetusega):

| Keel | Käsud |
|---|---|
| C | `fprintf(stdout, "1\n%s\n", s);`<br>`fflush(stdout);` |
| C++ | `cout << 1 << "\n" << s << endl;` |
| Java | `System.out.println("1");`<br>`System.out.println(s);`<br>`System.out.flush();` |
| Python | `sys.stdout.write("1\n{0}\n".format(s))`<br>`sys.stdout.flush()` |

**Testimine.** Iga testjuhu korral pannakse samaaegselt käima 4 sinu programmi eksemplari, 2 kirjutamise ja 2 lugemise jaoks. Protsessori aja ja mälumahu limiidid on antud kõigi programmi eksemplaride peale kokku. **Igasugune sihilik püüd edastada andmeid programmieksemplaride vahel väljaspool ettenähtud suhtlusprotokolli loetakse sohitegemiseks ja on alus diskvalifitseerimiseks.**

$B$ bitised mäluplokid initsialiseeritakse nullidega. Seejärel viiakse läbi kirjutamise ja lugemise opearsioonid mingis teostatavas järjekorras.

Kirjutamise operatsioonide ajal antakse kirjutavale eksemplarile ühe mäluploki hetkeolek ja väärtus, mis sinna tuleb salvestada. Sa võid eeldada, et kirjutatavad väärtused on valitud juhuslikult ühtlase jaotusega hulgast $0 \dots 2^M - 1$ ja on sõltumatud kõigest muust. Kui su programm on võimeline etteantud väärtuse salvestama, siis programmi poolt tagastatud mälu olek seatakse sama mäluploki uueks olekuks. Kui su programm ei saa väärtust salvestada, siis see mäluplokk ei osale enam kirjutamise operatsioonides.

Lugemise operatsioonide ajal antakse lugevale eksemplarile mäluploki olek, mis on saadud edukast kirjutamise operatsioonist. Kontrollitakse, kas sinu programmi poolt tagastatud väärtus on sama, mis pidi olema salvestatud kirjutamise operatsiooni käigus. Iga eduka salvestuse väljundit

kasutatakse lugemiseks täpselt ühe korra.

**Hindamine.** Igas testide grupis on sinu programmi skoor proportsionaalne kirjutatud väärtuste keskmise arvuga selle grupi testides. Täpsemalt öeldes, kui su programm on võimeline kirjutama keskmiselt $V$ väärtust testide grupi kohta, siis on programmi skooriks testgrupi väärtusest $100 \cdot V/P\%$, kus $P$ väärtus on esitatud allpool. Kui su programm tagastab lugemise operatsiooniga vale väärtuse, siis on kogu grupi eest saadav skoor null. Igasuguse muu vea korral loetakse selle testi käigus loetud väärtuste arv nulliks.

Testide grupid rahuldavad järgmiseid tingimusi:

1. (5 points) $B = 16$, $M = 8$, $P = 4.062445024495069624056$.
2. (5 points) $B = 32$, $M = 8$, $P = 12.264904841300964834177$.
3. (5 points) $B = 32$, $M = 16$, $P = 4.129591513707784802006$.
4. (5 points) $B = 64$, $M = 8$, $P = 30.039277894268828900030$.
5. (5 points) $B = 64$, $M = 16$, $P = 12.953148094217360432715$.
6. (5 points) $B = 64$, $M = 32$, $P = 4.073559788233661501537$.
7. (5 points) $B = 128$, $M = 8$, $P = 69.777892228928747548775$.
8. (5 points) $B = 128$, $M = 16$, $P = 34.731791275143635240976$.
9. (5 points) $B = 128$, $M = 32$, $P = 13.950788987705638908663$.
10. (5 points) $B = 128$, $M = 64$, $P = 4.039918210604800133907$.
11. (5 points) $B = 256$, $M = 8$, $P = 174.468047086071038511453$.
12. (5 points) $B = 256$, $M = 16$, $P = 82.222614151404177334554$.
13. (5 points) $B = 256$, $M = 32$, $P = 37.629382269769206488916$.
14. (5 points) $B = 256$, $M = 64$, $P = 14.263462282054140577686$.
15. (5 points) $B = 256$, $M = 128$, $P = 4.015569093893943430859$.
16. (5 points) $B = 512$, $M = 16$, $P = 204.746242127410346170221$.
17. (5 points) $B = 512$, $M = 32$, $P = 91.778595148073111539847$.
18. (5 points) $B = 512$, $M = 64$, $P = 39.2302792421459387126 21$.
19. (5 points) $B = 512$, $M = 128$, $P = 15.000000002167672268601$.
20. (5 points) $B = 512$, $M = 256$, $P = 4.005423277111055468876$.

Täiendavalt, kõigi testjuhtude korral kehtib $N \cdot B \leqslant 120\,000$, kus $N$ on maksimaalne kirjutamisoperatsioonide arv, mida su programmil võidakse käskida teha.

Võistluse ajal hinnatakse sinu lahendust väikese hulga testidega igast grupist. Pärast võistlust hinnatakse sinu viimati esitatud lahendust ja väikeste testide hulga peal parima punktisumma saanud lahendust suurema testide hulgaga ning neist kahest parim tulemus on sinu lõplikuks skooriks selle ülesande eest. **Skoor, mida sa näed CMS-i võistluste ajal, ei ole sinu lõplik skoor.** Sellise protseduuri eesmärgiks on sinu skoori täpsuse tõstmine. Hindamine täieliku testide hulgaga võistluse ajal oleks liiga aeglane.