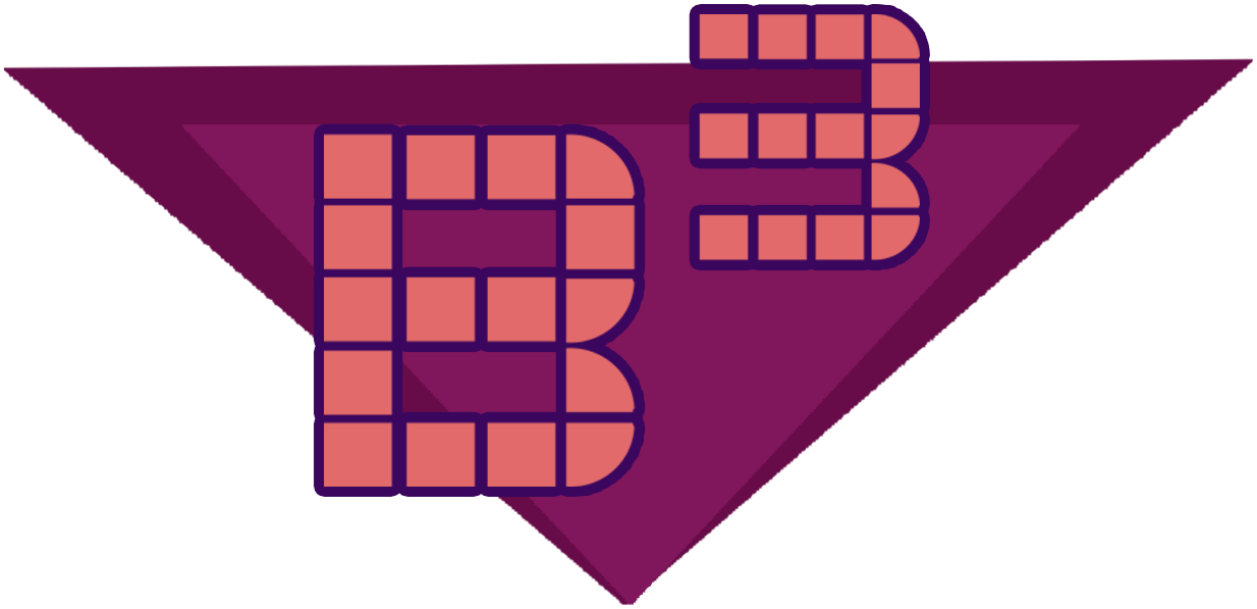


Bakonyi Bitfaragó Bajnokság
Döntő forduló
2022. November 19.
10:00-17:00

Kedves Bitfaragók!

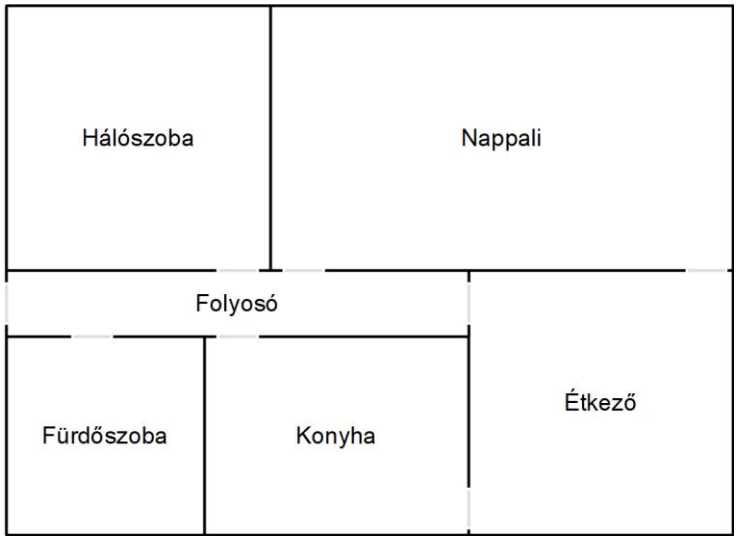
Köszöntünk a verseny döntő fordulóján. A mai nap során a feladat kettő részből áll. Az egyik részben a cél minél több kis feladat megoldása a rendelkezésre állók közül, a másikban pedig egy böngészős „szimuláció” készítése.

A téma

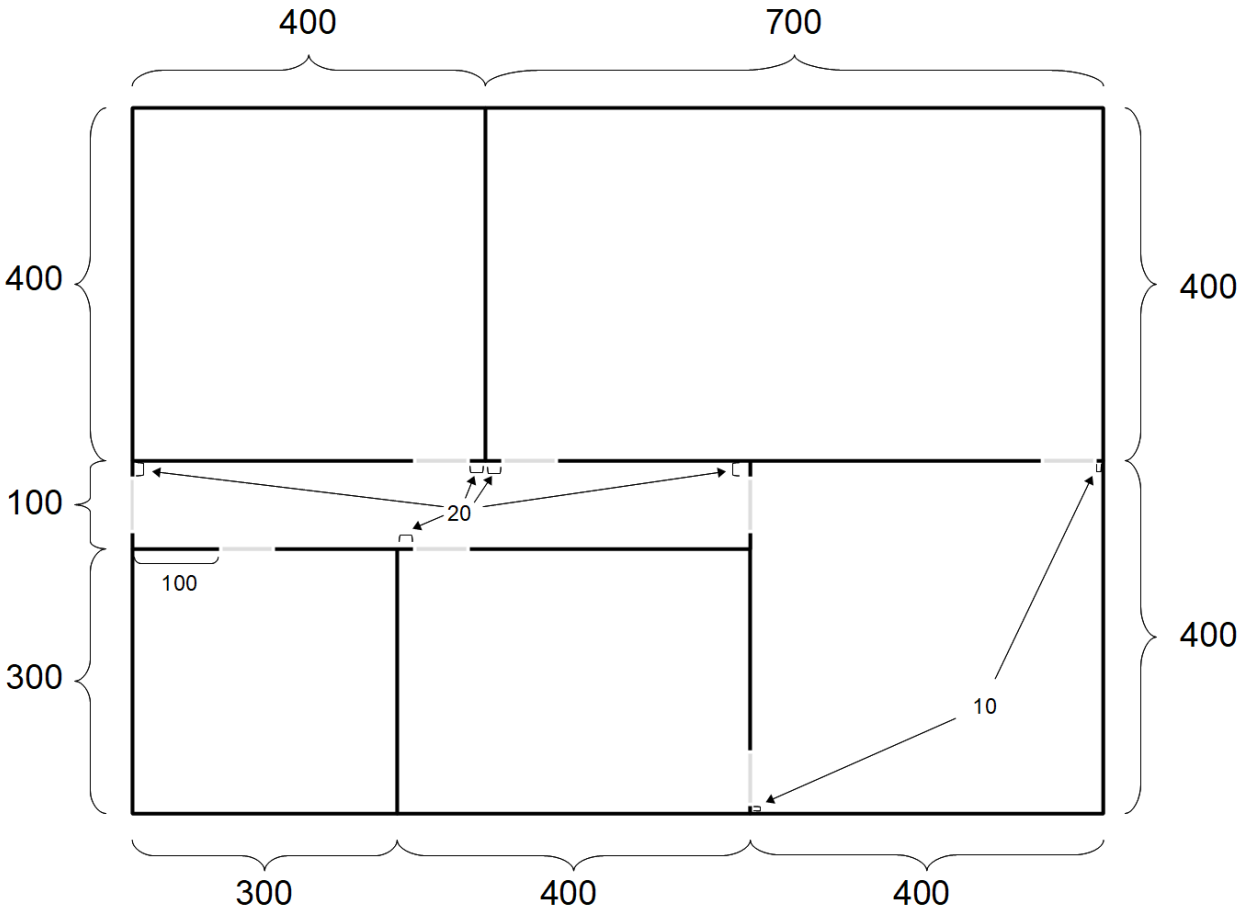
A döntő feladatai egy okosház működtetésével kapcsolatosak. A házat egy személy lakja, és a vezérlésbe szeretne okos eszközöket integrálni. A házban lesznek jelforrások (pl. villanykapcsolók, érzékelők), valamint vezérelhető egységek (pl. világítás, fűtés). A jelforrások és a vezérelt egységek közötti kapcsolatokat logikai áramkör írja le, ezzel definiálva a hálózat működését.

A ház

A ház alaprajza a szobákkal együtt látható:



A halvány szürkével jelölt szakadások a falakban az ajtók. A házon kívülre a folyosón keresztül lehet menni. A ház 11x7 m-es alapterületű. A szobák konkrét méretei, valamint az ajtók pozíciói a következő oldalon láthatóak. Minden mérőszám centiméterben van megadva. Minden ajtó 60 cm széles. A fürdő ajtaja 100 cm-re van a ház falától. A folyósón minden másik ajtó 20 cm-re van a csatlakozó szoba falától. Az étkezőből nyíló kettő extra ajtó 10-10 cm-re van a ház falától. Az egyszerűség kedvéért a feladat során a ház falának vastagságával nem foglalkozunk.
A feladatok során szükség lesz a házon belül pozíciók megadására. A pozíciót kettő egész számmal (x és y) adjuk meg, centiméterben mérve. A képen a bal alsó sarok (a fürdő külső sarka) felel meg a (0,0) pozíciónak, a jobb felső sarok (a nappali külső sarka) pedig az (1100,800) pozícióban található.



A falak vastagságával most külön nem foglalkozunk, és a pozíciók egész számok, így a falak tulajdonképpen kettő pozíció között vannak. Így a helyiségek közötti határok a következő módon alakulnak: a ház bal alsó sarka a (0,0) pozíció, ami már a fürdő része. Azért, hogy a fürdőszoba 300x300-as legyen, a helyiség jobb felső sarka a (299,299) pozíció lett. Ha az x érték eléri a 300-at, akkor az már nem a fürdőszobához, hanem a konyhához tartozik. Hasonlóan, ha az y érték éri el a 300-at, akkor az a pozíció a folyosón van. Hasonlóan a nappali a (400,400)-as pozíciótól az (1099,799)-ig tart, utána már szabadtér van, az (1100,800) pozíció már oda tartozik.

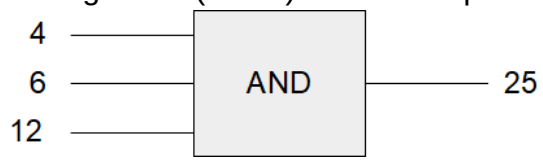
Összefoglalva az egyes szobákhoz tartozó pozíciók:

- Folyosó: (0,300) és (699,399) között,
- Konyha: (300,0) és (699,299) között,
- Étkező: (700,0) és (1099,399) között,
- Fürdőszoba: (0,0) és (299,299) között,
- Hálószoba: (0,400) és (399,799) között,
- Nappali: (400,400) és (1099,799) között.

A ház logikája

A ház okos részének vezérlését egy logikai áramkör vezérli, amely a jelforrásokból indulva, azokat kapukkal átalakítva irányítja a vezérelhető egységeket. A vezérelhető eszközöket egy-egy betű azonosítja, míg a jelforrások és a logikai kapuk kimenetei szám azonosítót kaptak. A logikai hálózatban a jelek egész számokat továbbítanak. A legtöbb esetben ez egy bináris (0 vagy 1) érték, de bizonyos jelek esetében (pl. mért hőmérséklet) lehet magasabb érték is.

A hálózatot két típusú összeköttetés segítségével definiáljuk. Az egyik csoport a logikai kapuk, amelyek egy vagy több bemenettel rendelkeznek, és 1 kimenetet generálnak. A kimenet értéke függ a bemenetek értékeitől, valamint a kapu típusától. Például definiálhatunk egy „és” kaput („AND” típussal), ami a bementére három jelet vár, ezek azonosítói 4, 6, és 12. Az „és” kaput a 25-ös azonosítóval hozzuk létre, így ennek a kimenete a 25-ös jel lesz. Mivel a kapu „és” típusú, ezért a kimenete akkor lesz 1, ha mindhárom bemenet értéke 1, minden más esetben 0. Ha valamelyik bemeneti jel nem bináris típusú, akkor az úgy kezeli, hogy a 0 értéken kívül minden mást igaznak (1-nek) tekint. A kapu sematikus rajza:

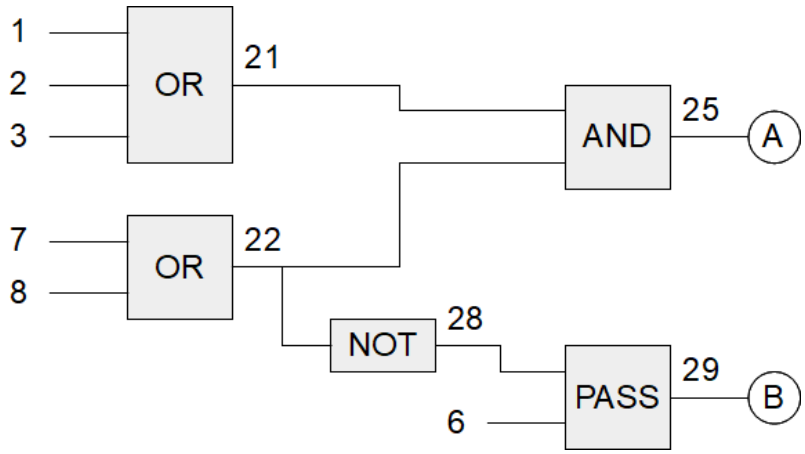


A használt rendszer több típusú kaput is támogat, amelyeket a feladatsor végén lévő táblázat foglal össze. Nem biztos, hogy a feladatok megoldása során mindegyikre szükség lesz, de ezeket lehet használni. Minden kapunak 1 kimeneti értéke van, ami lehet bináris vagy általános. A bemenetek száma változhat.

Az összeköttetések második típusa a vezérlés bekötése. Minden vezérelhető egységhez, amelyet irányítani akarunk a hálózattal, hozzá kell rendelni pontosan 1 jelet, amelyik azt vezérli. Bizonyos vezérelhető eszközökhöz (pl. világítás) bináris jel kell, míg másokhoz (például fűtés) nagyobb érték is rendelhető. Viszont minden egységet csak 1 jel vezérelhet. Ez a jel jöhet közvetlenül valamelyik forrásból, vagy lehet valamelyik kapu kimenete (vagyis bármilyen számmal azonosított jel).

A hálózatok esetében fontos, hogy nem lehet benne kör (egyik kapu kimenete sem tudja befolyásolni a saját bemeneteit, sem közvetlenül, sem közvetve).

Például tekintsük a hálózatot a lentebbi ábrán. Ez a következő működést valósítja meg: Az „A” eszköz akkor kapcsoljon be, ha az 1, 2, 3 jelek közül legalább az egyik 1-es, valamint a 7 és 8 jelek közül is legalább az egyik 1-es. Az „A” eszköz vezérlését így a 25-ös azonosítóval létrehozott „AND” kapu adja, ami két „OR” kapu kimeneteit kapja bemenetnek. A másik vezérelt eszköz, a „B” címkével rendelkező, egy általános szám értéket kap a bementére, ami a 6-os jel értékével egyezik meg, de csak akkor, ha az 1, 2, 3 jelek közül egyik sem igaz. Mivel a 22-es „OR” kapu már elkészült, ezért a „B” eszközhöz ez újra használható egy tagadás beiktatásával (mivel bármelyik jel szétbontható), aminek a kimenete aztán a „PASS” kapu első bemenetére kerül, és ez dönt arról, hogy a kapu kimenete 0 legyen, vagy a 6-os jelből érkező érték. Ennek a kapunak a 29-es jelű kimenete vezérli a „B” eszközt.



A ház tartalmaz alapértelmezett jelforrásokat (pl. villanykapcsoló, termosztát), de ezen felül lehet újabb szenzorokat is elhelyezni a feladatok megoldásai során (pl. hőmérséklet érzékelő, mozgás érzékelő). A vezérelhető eszközök adottak, azokhoz újat tenni nem lehet, de a feladatokban jellemzően nem kell mindet vezérelni. A beépített jelforrások, a letehető szenzorok, valamint a vezérelhető egységek részletezése a feladatsor végén található.

Feladatok

A döntő során az eredmény kettő összetevőből fog állni.
Egyrészt egy szerveren található több kis feladat leírása, amelyek megoldásaiért lehet pontokat gyűjteni. Ezek a feladatok tipikusan arról szólnak, hogy a lakásban a jelforrások értékeit logikai áramkörrel összekötve vezéreljük a vezérelhető egységeket a feladatban megadott szabályok szerint. Ezen feladatok megoldásához túl sok forráskód nem szükséges, csak a szerverrel való kommunikációt kell megvalósítani. A logikai áramkörök kézzel megtervezhetőek, de ez is hasonlít a programozásra. A kis feladatok megoldásait a szerver automatikusan ellenőrzi.
A feladat másik fele egy szimuláció elkészítése, amely a böngészőben fut. A szimuláció a ház logikájának működését ábrázolja úgy, hogy a logikai áramkör módosítható. Az ehhez készített megoldásokat a zsűri fogja értékelni.
A végeredményben a kis feladatokban elért pontszám és a szimulációra kapott pontszám azonos súllyal szerepel, mindkettő egyformán fontos. Következzenek a részletek.

Kis feladatok rendszere

A kis feladatok során egy központi rendszerrel kell kommunikálni, amely folyamatosan számon tartja majd az elkészült feladatokat. A kommunikáció távoli kérésekkel történik, JSON formátumú üzenetek küldésével. A feladatok különböző kérdéseket fognak tartalmazni, amit a szerver tesz fel, és neki kell megküldeni a választ. A kérdés mindig egy adott hálózat megtervezése lesz, a szervernek a hálózat leírását kell elküldeni.

A szervert a http://bitkozpont.mik.uni-pannon.hu/2022/ címen lehet elérni, ide kell majd a kéréseket küldeni. A fordulóra minden csapat külön kap egy azonosító kódot. Ezt a kódot minden üzenethez mellékelni kell majd, amit a szerver felé küldtök. Ennek pontos módját a későbbiekben részletezzük. A szerver három típusú kérést tud kiszolgálni, nézzük is sorba, de előtte még nézzük a kommunikáció általános módját.

Általános kommunikáció

A kérésekhez POST módszerrel paraméterként kötelező megadni a csapatnak a kódját:

```
{
        teamcode: „d08055dab5f72a650c0e”
}
```

A *teamcode* paramétert minden üzenethez mellékelni kell, ahol a kód értéke a csapat által külön kapott kód legyen. Ezt kell JSON-be kódolva hozzátenni az üzenethez. Az üzenetben ezen felül más paraméter is lehet.

Például az összes feladat lekérése:

```
xhttp.open("POST", " http://bitkozpont.mik.uni-pannon.hu/2022/gettasks.php", true);
xhttp.send(JSON.stringify({
        id: „all”,
        teamcode: „d08055dab5f72a650c0e”
}));
```

Minden, a szerver által küldött válaszüzenet egységes felépítéssel rendelkezik:

```
{
  "status": "success",
  "data": {...},
  "message": "",
  "hash": ""
}
```

Itt a status a sikerességet jelzi, minden esetben a success a jó. A data a válasz tényleges tartalma, ez lekérésenként változik. A message különböző üzeneteket tartalmazhat, elsősorban, ha valami nem jó. A hash néha üres, néha tartalmaz egy kódot, a helyzettől függően.

Minden feladat lekérése

A szerveren a következő címről lehet a feladatok listáját lekérdezni: http://bitkozpont.mik.uni-pannon.hu/2022/gettasks.php. A lekéréshez a következő adatok kellenek:

```
{
        id: „all”,
        teamcode: „d08055dab5f72a650c0e”
}
```

Itt az *id* az fixen az *„all”* értéket kapja.

A válasz üzenetben a *data* tartalmazza a feladatok listáját (*task_list*), és az azok közötti összefüggéseket (*dependencies*). A szervertől kapott listában a csapat

láthatja azt is, hogy melyik feladatok készítették már el, hogy melyik mennyi pontot ér, valamint, hogy melyik feladat melyik másikra épít.

A feladatok állapota 5 féle lehet, de ebből most csak az alábbi 4 fordul elő:

- OPEN: Ezek a nyitott feladatok, amiken a csapat dolgozhat, de megoldás még nem született.
- COMPLETED: Ezek a már sikeresen megoldott feladatok. Ezekkel már jellemzően nem kell foglalkozni, de természetesen vissza lehet térni rájuk.
- LOCKED: Azok a feladatok, amelyeknek az előzményeit még nem sikerült teljesíteni, így nem tekinthetőek meg.
- READONLY: Bizonyos feladatoknak több előfeltétele is van. Ezen feladatok esetében az előzmények 2/3-ának teljesítése (pl. 6-ból 4, vagy 4-ből 3) már elég ahhoz, hogy a feladat megtekinthető legyen. A szerver ilyen esetben már a válaszokat is ellenőrzi a feladathoz, azonban a teljesítést csak akkor fogadja el, ha már minden előzményt sikerült teljesíteni.

Egy feladat lekérése:

Egy feladat részletei szintén a http://bitkozpont.mik.uni-pannon.hu/2022/gettasks.php címen érhetőek el. A lekéréshez a következő adatok kellenek:

```
{
        id: 1,
        teamcode: „d08055dab5f72a650c0e”
}
```

Itt az *id* a konkrét feladat azonosítója legyen. Ha a feladat nem LOCKED, akkor a szerver közli az adatait:

```
{
   "status": "success",
   "data": {
      "attempt": 3,
      "description": "descriptions/t034314.html",
      "questions": [
         {
            "ID": "3",
            "question_id": "3",
            "question_type": "CREATE_NETWORK",
            "order": "1",
            "params": {
               "type": "BATHROOM_SENSOR_BASIC"
            }
         }
      ],
      "ID": 3,
      "level": 1,
      "stage": 2,
      "points": 1,
      "state": "COMPLETED"
```

```
  },
  "message": "",
  "hash": "8f1c8f0adb2ed79addc48cf61b20987070513003"
}
```

Itt a fontos rész a *„data”*-n belül van. Az *„attempt”* csak azt tartja számon, hogy hányszor kérte le a csapat ezt a feladatot, ez csak a szervernek fontos. Egyrészt fontos a *„description”* által tartalmazott leírás. Ez egy HTML oldal címét tartalmazza a szerveren, amit célszerű a weblapon megjeleníteni, vagy külön megnyitni, hiszen ez fogja mindig tárolni az adott feladathoz tartozó részletes leírást. A „*questions*” rész sorolja fel egy tömbben az egyes kérdéseket, amire fentebb látható egy példa. A kérdésnek van két azonosítója (ebből az ID kell), egy típusa, és egy sorrendje a feladaton belül. Az idei versenyen a „*questions*” rész mindig csak egy kérdést fog tartalmazni, aminek a típusa „CREATE_NETWORK” lesz, ezzel nem kell foglalkozni. A kérdéshez tartozó „*params*” a kérdés típusát tartalmazza, ez megint csak a szerver számára lesz fontos, hiszen mindig az adott feladatnak felel meg az értéke. A *„hash”* adat fontos, mert ezt majd a szervernek vissza kell küldeni.

A feladat leírását tartalmazó html fájl bizonyos feladatoknál tartalmazhat extra információt a feladatról, vagy példát a szerver által küldött kérdésre, és a megfelelő válaszra. Ezek nyilván arra vannak, hogy a kérdés és válasz felépítését egyértelműsítsék. A későbbi feladatok esetén már nincs ilyen, addigra már ismerős lesz a rendszer.

Válaszok elküldése

Amikor a feladat megoldásra került (minden kérdésre van válasz), akkor a válaszokat el kell küldeni a szervernek.

A válasz beküldéséhez a http://bitkozpont.mik.uni-pannon.hu/2022/answer.php címre kell kérést küldeni a következő paraméterekkel:

```
{
        id: 1,
        teamcode: „d08055dab5f72a650c0e”,
        original_data: {...},
        original_hash: „...”,
        answer_data: [...]
}
```

Itt az *„id”* a feladat azonosítója, a *„teamcode”* a csapat kódja. Az üzenetbe *„original_data”* néven bele kell tenni a feladat lekérése során megkapott teljes *„data”* részt, valamint *„original_hash”* néven a megkapott eredeit *„hash”* szöveget. Ezek nélkül a szerver nem fogja tudni megfelelően értékelni a válaszokat. A tényleges válaszok az *„answer_data”* részben vannak, ami egy tömbben tárol egy-egy választ minden kérdéshez, ilyen módon:

```
{
        id: ...,
        answer: ...
}
```

Az *„id”* a kérdés azonsítója (ami a kérdés leírásában az *„ID”*), az *„answer”* a válasz.

A fenti példa feladatra egy példa válasz üzenet tehát így néz ki (maga a megoldás nem helyes ehhez a feladathoz):


```
{
   "answer_data": [
      {
         "answer": {
            "additionalSources": [
               {
                  "fixvalue": 0,
                  "id": 10,
                  "position": {
                     "x": 78,
                     "y": 547
                  },
                  "sensortype": "TEMPERATURE"
               },
               {
                  "fixvalue": 19,
                  "id": 21,
                  "position": null,
                  "sensortype": "FIXVALUE"
               }
            ],
            "controlConnections": [
               {
                  "controlid": "D",
                  "inputid": 5
               },
               {
                  "controlid": "B",
                  "inputid": 30
               }
            ],
            "logicConnections": [
               {
                  "gatetype": "MIN",
                  "id": 30,
                  "inputids": [
                     7,
                     10,
                     21,
                  ]
               }
            ]
         },
         "id": "3"
      }
   ],
   "id": "3",
   "original_data": {
      "attempt": 3,
      "description": "descriptions/t034314.html",
```

```
      "ID": 3,
      "level": 1,
      "points": 1,
      "questions": [
         {
            "ID": "3",
            "order": "1",
            "params": {
               "type": "BATHROOM_SENSOR_BASIC"
            },
            "question_id": "3",
            "question_type": "CREATE_NETWORK"
         }
      ],
      "stage": 2,
      "state": "COMPLETED"
   },
   "original_hash": "8f1c8f0adb2ed79addc48cf61b20987070513003",
   "teamcode": " d08055dab5f72a650c0e"
}
```


A válaszokra a szerver egy üzenettel reagál, amiben leírja, hogy jónak találta-e a hálózat a tesztelés során, vagy nem. Amennyiben a hálózattal probléma van (pl. nem játszható végig a működése), akkor ezt is jelzi a válaszban. Továbbá, ha valamelyik tesztre rossz eredményt ad az elküldött hálózat, akkor visszaad egy ilyen tesztet, amiben megadja a jelforrások értékeit, az ezekre elvárt eredményt (a feladat leírás szerint), valamint a hálózat által adott eredményt.

Szimulációs feladat

A kis feladatok megoldása mellett a nap másik célja a feladatok során felépített hálózatok szimulációja böngészőben. Készíteni kell egy olyan alkalmazást, amely megjeleníti a ház alaprajzát, valamint benne a lakót, akivel lehet mászkálni a házban (illetve házon kívül is). A felületen jelenjenek meg a lakásban lévő alapértelmezett jelforrások és vezérelhető egységek (megjelenhetnek akár az alaprajzon is, de külön helyen is jó).

Lehessen szerkeszteni a ház logikai hálózatát (hozzáadni új forrásokat, valamint beállítani a kapcsolatokat). Lehessen a jelforrások értékeit állítani, valamint a lakóval mozogni a lakásban. A program folyamatosan szimulálja újra a logikai hálózatot, és mindig frissítse a vezérelhető eszközök által kapott jelet a hálózatnak és az aktuális jelforrásoknak megfelelően.

A programba legyenek beépítve a megoldott kis feladatokra adott helyes megoldások is, amiket lehessen betölteni, és ezáltal tesztelni.

Ennek a résznek az értékelését a zsűri fogja végezni. A verseny utolsó 1,5 órájában lehetőség lesz ezt a felületet bemutatni a zsűrinek (akár többször is, ha még sikerül menet közben javítani rajta). A zsűri a felületet értékelni fogja teljesség, megjelenés, ötletesség, használhatóság szempontjából.

A verseny eredménye

A verseny során a kapott pontszám nagy részét a sikeresen megoldott feladatok, valamint a szimulációra kapott értékelés teszik ki (egyenlő arányban), azonban az értékelés nem csak ennyiből áll. A verseny végén minden csapatnak tartani kell egy rövid prezentációt, amiben ismertetik a megoldásukat, a kihívásokat, a legszebb eredményeket, és ezt a prezentációt is értékeljük, valamint figyelembe vesszük az elkészített munka pontozásánál.

Támogatott kapu típusok

Típus	Megnevezés	Bemenetek száma	Kimenet értéke
NOT	Invertálás	1	Bináris: 1, ha a bemenet 0 0, ha a bemenet 1
OR	Vagy kapu	Tetszőleges	Bináris: 1, ha bármelyik bemenet 1 0, ha mindegyik bemenet 0
NOR	Nem vagy kapu (a vagy kapu inverze)	Tetszőleges	Bináris: 1, ha mindegyik bemenet 0 0, ha bármelyik bemenet 1
AND	És kapu	Tetszőleges	Bináris: 1, ha mindegyik bemenet 1 0, ha bármelyik bemenet 0
NAND	Nem és kapu (az és kapu inverze)	Tetszőleges	Bináris: 1, ha bármelyik bemenet 0 0, ha mindegyik bemenet 1
EQ	Egyenlő	2	Bináris: 1, ha a kettő bemenet megegyezik 0, ha a kettő bemenet eltér
NEQ	Nem egyenlő (az egyenlő inverze)	2	Bináris: 1, ha a kettő bemenet eltér 0, ha a kettő bemenet megegyezik
COMP	Összehasonlítás	2	Bináris: 1, ha az első bemenet kisebb a másodiknál 0, ha nem (egyenlőség esetén is 0)
PASS	Feltételes áteresztő	2	Általános (a második bemenettel megegyező) 0, ha az első bemenet 0 Minden más esetben a második bemenet értéke kerül a kimenetre
MIN	Legkisebb	Tetszőleges	Általános A kimenet a bemenetek közötti legkisebb értékkel egyezik meg.
MAX	Legnagyobb	Tetszőleges	Általános A kimenet a bemenetek közötti legnagyobb értékkel egyezik meg.

Beépített jelforrások

Ezek a jelforrások már szerepelnek a házban, az ezek által adott értékek bármikor felhasználhatóak.

Azonosító	Megnevezés	Jel értéke
1	Folyosó villanykapcsoló	Bináris: 1, ha fel van kapcsolva, 0 ha nem
2	Konyha villanykapcsoló	Bináris: 1, ha fel van kapcsolva, 0 ha nem
3	Étkező villanykapcsoló	Bináris: 1, ha fel van kapcsolva, 0 ha nem
4	Fürdőszoba villanykapcsoló	Bináris: 1, ha fel van kapcsolva, 0 ha nem
5	Hálószoba villanykapcsoló	Bináris: 1, ha fel van kapcsolva, 0 ha nem
6	Nappali villanykapcsoló	Bináris: 1, ha fel van kapcsolva, 0 ha nem
7	Termosztát	Általános egész érték (a kívánt fűtési hőmérséklet fokban)
8	Főkapcsoló	Bináris: 1, ha fel van kapcsolva, 0 ha nem
9	Gáz ár figyelő	Általános egész érték (a fűtéshez használt gáz aktuális ára)

Újonnan elhelyezhető jelforrások

Ha új forrást helyezünk el, az ugyanolyan jelforrásként fog szerepelni a hálózatban, mint a beépített források. Új jelforrás elhelyezéséhez a következő adatokat kell megadni:

- Azonosító: egész szám, amelynek értéke legalább 10. Két egyforma azonosító nem lehet, de nem kötelező, hogy sorban kövessék egymást. Lehet bármilyen egész szám 10 és 100000 között.
- Pozíció: egy x és y koordináta a jelforrás helyzetéhez, amennyiben szükséges. Ha a forrásnak nem kell pozíció, akkor legyen ***null***.
- Típus: a jelforrás típusa (lásd később)
- Fix érték: a jelforrás által kibocsájtott fix érték, amennyiben van ilyen. Ha nincs, akkor itt legyen 0.

Az új jelforrások típusai:

- Fix érték („FIXVALUE”): ekkor a pozíció nem kell, a fix érték adattag adja meg a fix értéket, amit a forrás küld.
- Mozgásérzékelő („LOCATION”): a pozíció alapján tudja, melyik szobában van. 1-es értéket küld, ha a lakó ugyanabban a szobában van, 0-s értéket, ha nem. Csak a lakásba rakható, a szabadba nem. Fix értéke nincs.
- Hőmérő („TEMPERATURE”): a pozíció alapján tudja, melyik szobában van. A szoba aktuális hőmérsékletét adja vissza. Szabadba is rakható, ekkor a kinti hőmérsékletet méri. Fix értéke nincs.
- Fény érzékelő („LIGHTLEVEL”): a pozíció alapján tudja, melyik szobában van. A szobában lévő aktuális fény szintjét küldi jelként. Szabadba is rakható, ekkor a kinti fényerőt méri. Fix értéke nincs.
- Távolság érzékelő („PROXIMITY”): pozíció kell. Azt mondja meg, hogy a lakó mennyire van közel a szenzorhoz, és ezt az értéket küldi jelként. Maximális hatótávja 3 méter, annál messzebbről nem érzékel. A küldött jel nem a

távolság, hanem a közelség (3 méter mínusz a távolság): ha a távolság 85 cm, akkor a jel értéke 215; ha a távolság 198 cm, akkor a jel értéke 102; stb. Az alkalmazott technológia miatt a távolságot falon keresztül is méri. Fix értéke nincs.

Vezérelhető eszközök

Azonosító	Megnevezés	Várt jel
A	Folyosó világítás	Bináris: 1 esetén világít, 0 esetén nem
B	Konyha világítás	Bináris: 1 esetén világít, 0 esetén nem
C	Étkező világítás	Bináris: 1 esetén világít, 0 esetén nem
D	Fürdőszoba világítás	Bináris: 1 esetén világít, 0 esetén nem
E	Hálószoba világítás	Bináris: 1 esetén világít, 0 esetén nem
F	Nappali világítás	Bináris: 1 esetén világít, 0 esetén nem
G	Folyosó fűtés	Általános egész: a kívánt fűtési hőmérséklet fokban
H	Konyha fűtés	Általános egész: a kívánt fűtési hőmérséklet fokban
I	Étkező fűtés	Általános egész: a kívánt fűtési hőmérséklet fokban
J	Fürdőszoba fűtés	Általános egész: a kívánt fűtési hőmérséklet fokban
K	Hálószoba fűtés	Általános egész: a kívánt fűtési hőmérséklet fokban
L	Nappali fűtés	Általános egész: a kívánt fűtési hőmérséklet fokban
M	Zene a lakásban	Bináris: 1 esetén szól a zene, 0 esetén nem
N	Riasztó	Bináris: 1 esetén aktív a riasztó, 0 esetén nem