C++ TILIDA MANTIQIY IFODALARNI ALGORITMINI TUZISH.

Ashirmatova Shaxrizoda Sharifjon qizi
Nizomiy nomidagi O'zbekiston Milliy Pedogogika Universiteti talabasi.

https://doi.org/10.5281/zenodo.20235856

Annaotatsiya. *C++ tilida mantiqiy ifodalarni robototexnika faniga moslab tushuntirish uchun, dastlab robototexnikada mantiqiy ifodalar qanday ishlatilishini ko'rib chiqamiz. Robototexnikada mantiqiy ifodalar va operatorlar robotlarning xatti-harakatlarini boshqarishda, masalan, sensorlardan olingan ma'lumotlar asosida qaror qabul qilishda, harakatlarni boshqarishda va xatoliklarni aniqlashda ishlatiladi. Robototexnikada, mantiqiy ifodalar va operatorlar robotning qarorlar qabul qilish jarayonini avtomatlashtirishda muhim rol o'ynaydi.*

Tayanch so'zlar: *Robototexnika, mexatronika, robot, sensor, aktuator, Arduino, dasturlash, sun'iy intellekt.*

C++ tilida mantiqiy ifodalarni algoritmini tuzish uchun, dastlab mantiqiy operatorlar va ularning ishlash printsiplari haqida ma'lumot berib o'tish zarur. C++ dasturlash tilida, mantiqiy ifodalar uchun asosan quyidagi operatorlar ishlatiladi:

1. **AND (va)** — &&: Agar ikkala shartning ham qiymati haqiqat bo'lsa, natija haqiqat bo'ladi.
2. **OR (yoki)** — ||: Agar kamida bitta shart haqiqat bo'lsa, natija haqiqat bo'ladi.
3. **NOT (emass)** — !: Shartning teskari qiymatini olish.

Mantiqiy operatorlar yordamida sensorlar tomonidan olingan ma'lumotlarga asoslanib robotning xatti-harakatlari belgilanishi mumkin. Bu operatorlar robototexnikada ko'plab muhim vazifalarni bajarishda ishlatiladi, masalan, harakatni boshqarish, to'qnashuvlarni oldini olish, va muhitdagi o'zgarishlarga moslashish.

Mantiqiy Operatorlar (Logical Operators)

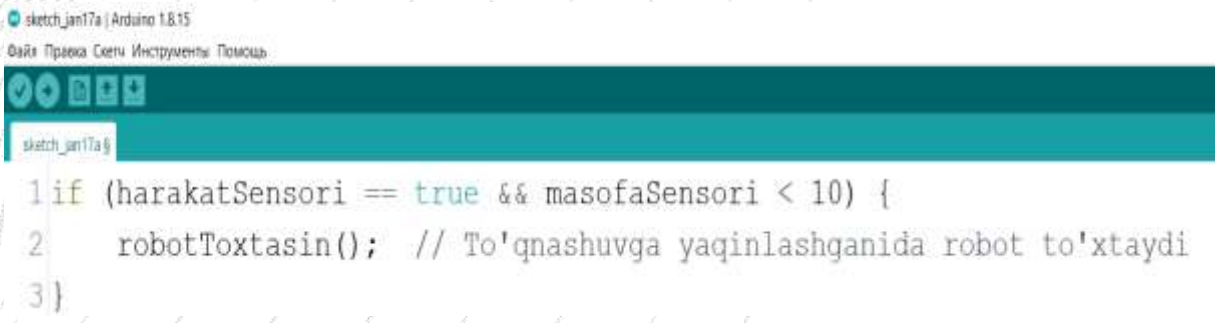
Mantiqiy operatorlar ikki yoki undan ortiq mantiqiy ifodalarni solishtirish va ulardan bitta natija (true yoki false) olish uchun ishlatiladi. C++ tilida quyidagi asosiy mantiqiy operatorlar mavjud:

1. AND (&&):

1.1 Har ikkala shart to'g'ri bo'lsa, natija true bo'ladi.

1.2 **Robototexnikada qo'llanilishi**: Agar ikki sensor bir vaqtning o'zida faollashsa, robot biror harakatni bajarishi mumkin. Masalan, agar harakat sensori va masofa sensori bir vaqtning o'zida "true" bo'lsa, robot to'qnashuvdan qochish uchun to'xtaydi.

Misol:

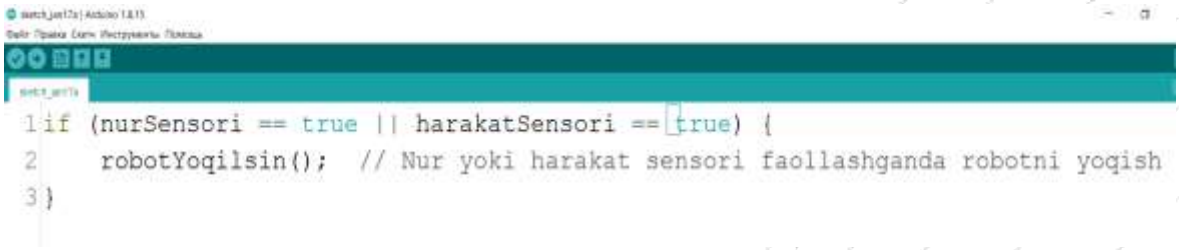


2. OR (||):

2.1 Agar kamida bitta shart to'g'ri bo'lsa, natija true bo'ladi.

- **2.2 Robototexnikada qo‘llanilishi**: Agar robot bir nechta sensorlardan foydalansa va ulardan biri faollashsa, robotning harakatini boshqarish mumkin. Masalan, agar nur sensori yoki harakat sensori faollashsa, robotni yoqish yoki yo‘lni aniqlash.

Misol:



```
if (nurSensori == true || harakatSensori == true) {
    robotYoqilsin();  // Nur yoki harakat sensori faollashganda robotni yoqish
}
```

3. NOT (!):

3.1 Shartning teskari qiymatini qaytaradi: agar shart true bo‘lsa, false qaytaradi va aksincha.

3.2 **Robototexnikada qo‘llanilishi**: Agar biror sensor o‘chirilgan bo‘lsa, robot boshqa amallarni bajarishi mumkin. Masalan, agar harakat sensori "false" bo‘lsa, robot harakatni davom ettirishi kerak.

Misol:

```
if (!(harakatSensori == true)) {
  robotHarakatiDavomEtsin();  // Agar harakat sensori faollashmagan bo'lsa, robot harakatni davom ettiradi
}
```

Shart Operatorlari (Conditional Operators)

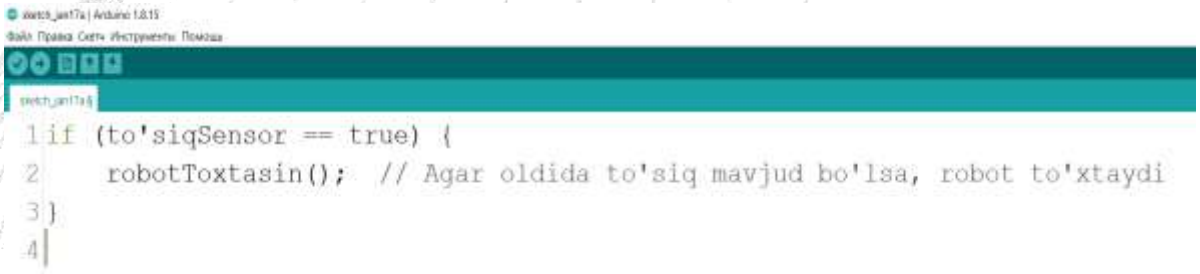
C++ tilida shart operatorlari yordamida mantiqiy ifodalar bo‘yicha qarorlar qabul qilinadi. Asosiy shart operatorlari:

1. if operatori:

1.1 Agar shart true bo‘lsa, blokdagi kod bajariladi.

1.2 **Robototexnikada qo‘llanilishi**: Agar sensorda biror o‘zgarish bo‘lsa, masalan, robotning oldida to‘siq mavjud bo‘lsa, shart operatori yordamida robotni to‘xtatish yoki boshqa harakatni bajarish mumkin.

Misol:



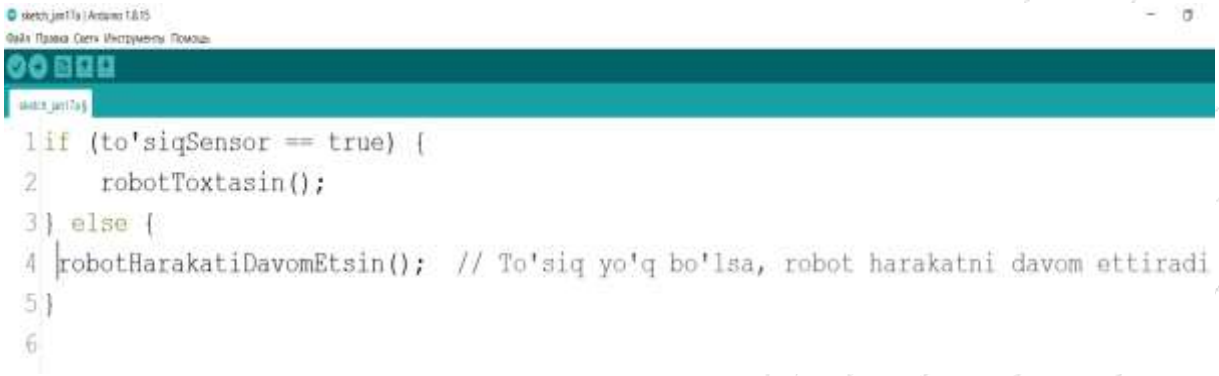
```
if (to'siqSensor == true) {
    robotToxtasin();  // Agar oldida to'siq mavjud bo'lsa, robot to'xtaydi
}
```

2. else operatori:

2.1 Agar if sharti false bo‘lsa, else blokidagi kod bajariladi.

2.2 **Robototexnikada qo‘llanilishi**: Agar shart bajarilmasa, robotning boshqa harakatini bajarish mumkin. Masalan, agar to‘siq sensori faollashmasa, robot oldinga harakat qiladi.

Misol:



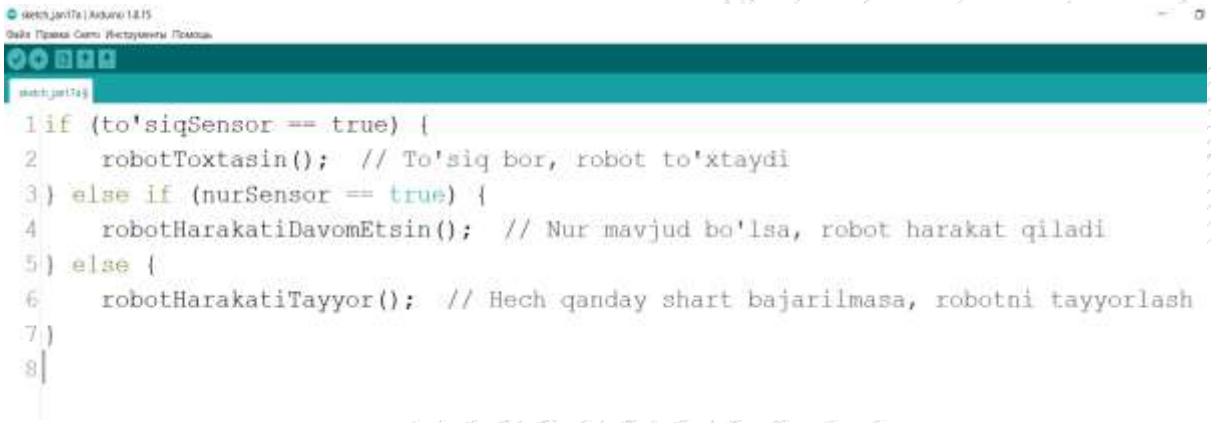
```
if (to'siqSensor == true) {
    robotToxtasin();
} else {
robotHarakatiDavomEtsin();  // To'siq yo'q bo'lsa, robot harakatni davom ettiradi
}
```

3. else if operatori:

3.1 Bir nechta shartlarni tekshirish uchun ishlatiladi.

3.2 **Robototexnikada qo‘llanilishi**: Agar robotning oldida to‘siq mavjud bo‘lsa, to‘xtatish, agar nur sensori ishlayotgan bo‘lsa, harakatni davom ettirish.

Misol:



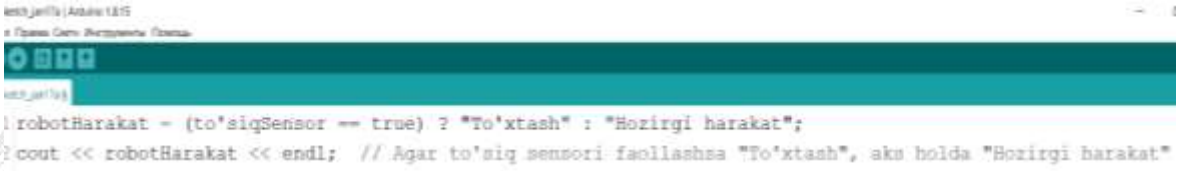
```
if (to'siqSensor == true) {
    robotToxtasin();  // To'siq bor, robot to'xtaydi
} else if (nurSensor == true) {
    robotHarakatiDavomEtsin();  // Nur mavjud bo'lsa, robot harakat qiladi
} else {
    robotHarakatiTayyor();  // Hech qanday shart bajarilmasa, robotni tayyorlash
}
```

4. Ternary operatori (?:):

4.1 Bu qisqargan shakldagi shart operatoridir. Agar biror shart bajarilsa, birinchi qiymatni qaytaradi, aks holda ikkinchi qiymatni qaytaradi.

4.2 **Robototexnikada qo‘llanilishi**: Agar robotning ma'lum bir shartga asoslangan xatti-harakatini aniqlash kerak bo‘lsa, ternary operatoridan foydalanish mumkin.

Misol:



```
robotHarakat = (to'siqSensor == true) ? "To'xtash" : "Hozirgi harakat";
cout << robotHarakat << endl;  // Agar to'siq sensori faollashsa "To'xtash", aks holda "Hozirgi harakat"
```

Robototexnikada mantiqiy operatorlar va shart operatorlarining qo‘llanilishi

Robototexnikada, mantiqiy va shart operatorlari sensorlardan olingan ma'lumotlarga asoslangan qarorlar qabul qilishda, robotni boshqarish va muhitga moslashishda muhim o‘rin tutadi. Ular robotning harakatini, javobini va tizimning xavfsizligini ta'minlashda yordam beradi.

Misollar:

1. **To‘qnashuvni oldini olish**: Agar robotning oldida to‘siq mavjud bo‘lsa, harakatni to‘xtatish uchun mantiqiy operatorlardan foydalaniladi.

2. **Harakatni boshqarish**: Agar nur sensori nur darajasini o‘lchasa, robot harakatini o‘zgartirishi mumkin. Masalan, nur darajasi yuqori bo‘lsa, robot yuqori tezlikda harakatlanishi mumkin.

3. **Xavfsizlik tizimi**: Agar xavfli holat yuzaga kelsa, masalan, robot ortga ketayotganda to‘siqni sezsa, robotning xavfsizlik tizimi mantiqiy ifodalar yordamida tizimni to‘xtatadi.

C++ tilidagi mantiqiy operatorlar va shart operatorlari robototexnikada sensorlardan olingan ma'lumotlarga asoslanib qarorlar qabul qilish va robotning harakatini boshqarishda muhim ahamiyatga ega. Ular robotga real vaqt shartlariga moslashish imkonini beradi va tizimning samaradorligini oshiradi.

Algoritm tuzish va amaliy misolini ishlab chiqish:

Algoritm tuzish — bu biror masalani yechish uchun ketma-ket bajariladigan amallarni tartib bilan keltirish jarayonidir. Robototexnikada algoritmlar robotning ma'lum bir vazifani bajarishini ta'minlash uchun ishlatiladi. Masalan, robotning harakati, sensorlardan olingan ma'lumotlarga asoslanib, qarorlar qabul qilish yoki muhitga moslashish uchun algoritmlar tuziladi.

Algoritmni yaratishda quyidagi bosqichlarni o‘z ichiga olishimiz mumkin:

1. **Muammoni aniqlash**: Masalani yoki vazifani aniq belgilash.
2. **Kirish ma'lumotlarini aniqlash**: Robotga kirish (input) ma'lumotlari kerak bo‘ladi (sensorlar, signal va hokazo).
3. **Chiqish ma'lumotlarini aniqlash**: Robotning chiqish (output) ma'lumotlarini belgilash (masalan, harakatlar).
4. **Amallarni aniqlash**: Bajariladigan amallarni (mantiqiy operatorlar, ifodalar, va boshqalar) aniqlash.
5. **Algoritmni yozish**: Amallarni ketma-ket tartibda yozish.
6. **Amaliyotga tatbiq etish**: Algoritmni amaliy robototexnika tizimiga qo‘llash.

Masala: Robot oldinga harakatlanib, to‘qnashuvlarni oldini olish kerak. Agar oldinda to‘siq bo‘lsa, robot to‘xtashi yoki yo‘nalishini o‘zgartirishi kerak.

Algoritm:

1. **Kirish ma'lumotlari**:

-oldingaHarakat (true yoki false)

-to‘siqSensori (true yoki false)

2. **Chiqish ma'lumotlari**:

-Robotning holati: "harakatlanish", "to‘xtash", yoki "yo‘nalishni o‘zgartirish".

3. **Amallar**:

-Agar to‘siqSensori = true bo‘lsa: Robot to‘xtasin.

-Agar to‘siqSensori = false bo‘lsa va oldingaHarakat = true bo‘lsa: Robot oldinga harakat qilsin.

-Agar to‘siqSensori = true va robotning yo‘nalishi to‘g‘ri bo‘lmasa: Robot yo‘nalishini o‘zgartirsin.

Algoritmning C++ tilidagi kod shakli:

```cpp
#include <iostream>
using namespace std;

int main() {
    // Kirish ma'lumotlari
    bool oldingaHarakat = true; // Robot oldinga harakat qilmoqda
    bool to'siqSensori = false; // To'siq yo'q

    // Robotning holatini tekshirish
    if (to'siqSensori == true) {
        cout << "Robot to'xtadi! To'siq bor." << endl;
    } else if (oldingaHarakat == true) {
        cout << "Robot oldinga harakat qilmoqda." << endl;
    } else {
        cout << "Robot harakatni to'xtatdi yoki boshqa harakat bajarilmoqda." << endl;
    }

    // Yo'nalish o'zgartirish
    if (to'siqSensori == true) {
        cout << "Robot yo'nalishini o'zgartirdi." << endl;
    }

    return 0;
}
```

Algoritmning ish faoliyati:

1. **Boshlanish**: Dastlab robot oldinga harakat qiladi va to‘siq sensori o‘chirilgan.
2. **To‘siq borligini tekshirish**: Agar to‘siq sensori faollashsa, robot to‘xtaydi.
3. **Harakatni davom ettirish**: Agar to‘siq mavjud bo‘lmasa, robot oldinga harakat qiladi.
4. **Yo‘nalish o‘zgartirish**: Agar robot oldinga harakat qilayotgan bo‘lsa, lekin to‘siq mavjud bo‘lsa, robot yo‘nalishini o‘zgartiradi.

Algoritmni kengaytirish: Misolni kengaytirish uchun robotga yanada murakkab harakatlar kiritish mumkin, masalan, robotning aylanib harakat qilishini ta'minlash, yoki boshqa sensorlar bilan bog‘lash.

Kengaytirilgan algoritm:Agar robotning oldida to‘siq mavjud bo‘lsa, u o‘z yo‘nalishini 90 daraja burib, yana oldinga harakat qiladi. Agar yana to‘siq paydo bo‘lsa, yana burilish amalga oshiriladi.

Kengaytirilgan C++ kodi:

```cpp
#include <iostream>
using namespace std;

int main() {
    // Kirish ma'lumotlari
    bool oldingaHarakat = true; // Robot oldinga harakat qilmoqda
    bool to'siqSensori = false; // To'siq yo'q
    bool yo'nalishOzgardi = false; // Yo'nalish o'zgarganini tekshirish

    // Robotning harakatini boshqarish
    while (oldingaHarakat) {
        if (to'siqSensori == true) {
            cout << "To'siq bor, robot to'xtadi!" << endl;
            cout << "Robot yo'nalishini o'zgartirdi." << endl;
            yo'nalishOzgardi = true;  // Yo'nalish o'zgardi
            to'siqSensori = false; // To'siq sensorini qaytadan tekshirish uchun
        }
```

```
18
19          if (yo'nalishOzgardi == true) {
20              cout << "Robot 90 daraja o'girildi va yana oldinga harakat qilmoqda." << endl;
21              yo'nalishOzgardi = false;  // Yo'nalishni o'zgartirish tugaydi
22          }
23
24          // Agar to'siq yo'q bo'lsa, robot harakatini davom ettiradi
25          if (to'siqSensori == false) {
26              cout << "Robot oldinga harakat qilmoqda." << endl;
27          }
28      }
29
30      return 0;
31  }
32
```

Bu misolda robotning to‘qnashuvlarni oldini olish uchun algoritm tuzildi. Dastlabki oddiy algoritm robotning oldida to‘siq bo‘lsa, to‘xtash va harakatni davom ettirishni ko‘rsatadi.Keyinchalik, bu algoritmni yanada murakkablashtirish orqali robotning harakatini dinamik boshqarish imkonini beradigan kengaytirilgan tizim yaratildi.

C++ tilida mantiqiy ifodalarni ishlatib kod yozish

C++ tilida mantiqiy ifodalar (logical expressions) yordamida dasturlarni yaratish robototexnikada juda muhim, chunki ular robotning turli holatlariga qarab qarorlar qabul qilishini ta'minlaydi. Mantiqiy ifodalar odatda **AND**, **OR**, va **NOT** operatorlari yordamida ishlaydi.

1. **AND (&&)**: Agar ikkala shart ham to‘g‘ri bo‘lsa, natija true bo‘ladi.
2. **OR (||)**: Agar kamida bitta shart to‘g‘ri bo‘lsa, natija true bo‘ladi.
3. **NOT (!)**: Agar shart false bo‘lsa, natija true bo‘ladi.

Misol: Robot xavfsizligini ta'minlash uchun, quyidagi shartlarni bajarish kerak:

-Agar robotning batareyasi to‘liq zaryadlangan bo‘lsa (true), va sensorlar tizimi to‘g‘ri ishlayotgan bo‘lsa (true), robot harakatlanishi mumkin.

-Agar batareya zaryadlanmagan bo‘lsa yoki sensorlar tizimi ishlamasa (false), robot harakatlanmaydi.

Bu masalani C++ tilida mantiqiy ifodalar yordamida yechamiz.

Algoritm:

1. Batareyani va sensorlarni tekshirish.
2. Agar ikkala shart ham to‘g‘ri bo‘lsa, robot harakatlanadi.
3. Aks holda, robot to‘xtaydi.

C++ kodi:

```
#include <iostream>
using namespace std;

int main() {
    // Kirish ma'lumotlari
    bool batareyaTo'liqZaryadlangan = true; // Batareya zaryadi to'liq
    bool sensorlarTizimiTo'g'ri = false;   // Sensorlar tizimi ishlamayapti

    // Mantiqiy ifoda: Agar batareya to'liq zaryadlangan va sensorlar tizimi ishlayotgan bo'lsa, robot harakatlanadi
    if (batareyaTo'liqZaryadlangan && sensorlarTizimiTo'g'ri) {
        cout << "Robot harakatlanmoqda." << endl;
    } else {
        cout << "Robot harakatlanmayapti: Batareya yoki sensorlar tizimida muammo mavjud." << endl;
    }
    return 0;
}
```

Izohlar: **batareya to‘liq zaryadlangan && sensorlarTizimiTo‘g‘ri**: Bu yerda **AND operatori (&&)** ishlatilgan. Bu ifoda faqatgina ikkala shart ham **true** bo‘lsa, **true** qiymatini qaytaradi. Agar **batareya To‘liq Zaryadlangan** va **sensorlar Tizimi To‘g‘ri** ikkalasi ham true bo‘lsa, robot harakatlanadi. Agar biri yoki ikkalasi ham false bo‘lsa, robot harakatlanmaydi. Agar batareya To‘liq Zaryadlangan = true va sensorlarTizimiTo‘g‘ri = true bo‘lsa, ekranda quyidagi natija ko‘rsatiladi: Robot harakatlanmoqda.

Agar sensorlar Tizimi To‘g‘ri = false bo‘lsa, ekranda quyidagi natija ko‘rsatiladi: Robot harakatlanmayapti: Batareya yoki sensorlar tizimida muammo mavjud.

Kengaytirilgan misol: Robot harakati to‘liq xavfsiz bo‘lishi uchun, quyidagi holatlar yuzaga kelganda ogohlantirish tizimi ishga tushishi kerak: Agar batareya darajasi juda past bo‘lsa (batareya <= 20%), robot ogohlantirish yuboradi. Agar sensorlar tizimida nosozlik bo‘lsa (sensorlarTizimiTo‘g‘ri = false), robot ham ogohlantirish yuboradi. Agar ikkala holat ham mavjud bo‘lsa, robot ham batareya pastligi, ham sensor tizimi muammosi haqida ogohlantiradi.

C++ kodi:

```cpp
#include <iostream>
using namespace std;
int main() {
    // Kirish ma'lumotlari
    int batareya = 15; // Batareya darajasi
    bool sensorlarTizimiTo'g'ri = false;   // Sensorlar tizimi ishlamayapti
    // Mantiqiy ifodalar yordamida ogohlantirish
    if (batareya <= 20 && !sensorlarTizimiTo'g'ri) {
        cout << "Ogohlantirish! Batareya past va sensorlar tizimi ishlamayapti!" << endl;
    } else if (batareya <= 20) {
        cout << "Ogohlantirish! Batareya past!" << endl;
    } else if (!sensorlarTizimiTo'g'ri) {
        cout << "Ogohlantirish! Sensorlar tizimi ishlamayapti!" << endl;
    } else {
        cout << "Robot xavfsiz ishlamoqda." << endl;
    }
    return 0;
}
```

Natija: Agar batareya = 15 va sensorlar Tizimi To‘g‘ri = false bo‘lsa, ekranda quyidagi natija ko‘rsatiladi:

Ogohlantirish! Batareya past va sensorlar tizimi ishlamayapti!

Agar batareya = 15 va sensorlar Tizimi To‘g‘ri = true bo‘lsa, quyidagi natija ko‘rsatiladi:

Ogohlantirish! Batareya past!

Agar batareya = 50 va sensorlar Tizimi To‘g‘ri = false bo‘lsa, quyidagi natija ko‘rsatiladi:

Ogohlantirish! Sensorlar tizimi ishlamayapti!

Agar batareya = 50 va sensorlar Tizimi To‘g‘ri = true bo‘lsa, quyidagi natija ko‘rsatiladi:

Robot xavfsiz ishlamoqda.

Mantiqiy ifodalar robototexnika tizimlarida robotning turli holatlariga qarab qarorlar qabul qilishda juda muhim vositadir. Bu misollarda batareya holati va sensor tizimi holati kabi shartlarga asoslanib robotning harakatini yoki xavfsizlik ogohlantirishlarini boshqarish ko‘rsatilgan. C++ dasturlash tilida mantiqiy operatorlar yordamida bunday tizimlarni yaratish osonlashadi va robotning muhitga moslashishini ta'minlaydi.

Xulosa:

"C++ tilida mantiqiy ifodalarni algoritmini tuzish" mavzusi dasturlashda mantiqiy fikrlashni rivojlantirishga qaratilgan asosiy tushunchalarni o‘z ichiga oladi.

C++ dasturlash tilida mantiqiy ifodalar yordamida murakkab shartli va takrorlanuvchi amallarni bajarish, hamda turli xatti-harakatlarni boshqarish mumkin.

Mantiqiy ifodalar - dasturda shartli qarorlar chiqarish va harakatlarni boshqarish uchun ishlatiladi. C++ da mantiqiy ifodalar if, else if, else, switch-case, va ternary operatori kabi turli shakllarda mavjud. Bu ifodalar yordamida dasturga kiritilgan shartlar asosida turli holatlar uchun mos javoblar chiqarish mumkin. Shart operatorlari (if, else, switch) yordamida kodda turli shartlarni tekshirib, ular bo‘yicha tegishli amallar bajariladi.

Masalan, son musbat yoki manfiyligini tekshirishda yoki foydalanuvchidan olingan ma'lumotlar asosida qaror qabul qilishda ishlatiladi. Mantiqiy operatorlar (masalan, && (va), || (yoki), ! (inkor)) yordamida murakkab shartlarni yaratish mumkin. Bu operatorlar bir nechta shartni birlashtirib, dasturga yanada aniqroq va kengroq shartlar qo‘shish imkonini beradi.

Dastur algoritmlarini yaratishda mantiqiy ifodalar va shart operatorlari yordamida masalani tahlil qilish va yechimlarni amalga oshirish samarali bo‘ladi. Har bir holat uchun tegishli qarorlarni ko‘rsatadigan algoritmni yaratishda bu operatorlar keng qo‘llaniladi.

Robototexnika va Arduino platformasida mantiqiy ifodalar va shart operatorlari juda muhim ahamiyatga ega. Masalan, sensorlardan olingan ma'lumotlarni qayta ishlash, turli shartlar asosida robotning harakatlarini boshqarish, yoki Arduino tizimida sensorlar orqali xatti-harakatlarni sozlashda mantiqiy ifodalar ishlatiladi. Algoritm tuzishda mantiqiy ifodalar dasturchilarga qadam-baqadam yechimlarni yaratish imkonini beradi. Bu esa kodni o‘qish va tushunishni osonlashtiradi va xatoliklarni tez aniqlashga yordam beradi.

FOYDALANILGAN ADABIYOTLAR

1. X.N.Nazarov ―Robototexnika asoslari‖. TDTU., Toshkent:, 2005 y.
2. “Робототехника для детей и родителей” С.А. Филиппов, Санкт-Петербург “Наука” 2010. - 195 с. 12
3. Бокселл Дж. Б78 Изучаем Arduino. 65 проектов своими руками. — СПб.: Питер, 2017. . — 400 с.
4. С. Монк Программируем Arduino. Профессиональная работа со скетчами . — СПб.: Питер, 2017.

Axborot manbaalari

1. http://www.robotics.uc.edu
2. http://www.robotics.utexas.edu./rrg
3. http://www.lib.tpu.ru/fulltext2/m/2012/m85.pdf