**Mid Semester Exam**

**Nama:** Isa Aulia Almadani

**Nim: K3523038**

**Rumusan masalah:** Pada program yang saya buat kali ini, saya ingin menjelaskan beberapa problem dan challenge yang harus dilewati si robot agar bisa keluar dari yang namanya *robotto meiro chālenji*. Dalam bahasa jepang, arti nama tersebut adalah permainan labirin yang menggunakan robot sebagai objeknya.

Terdapat lima challenge yang saya buat, pertama adalah ‘Simple labirin’. Pada rintangan ini aturanya adalah tidak boleh menghitung langkah dan memakan beacon. Jadi bagaimana cara si robot bisa menuju ke jalan keluar walaupun nantinya sempat menemukan jalan buntu. Terkahir, bagaimana cara si robot memindahkan beacon untuk memasuki challange kedua.

Challenge yang kedua, yaitu ‘Turn on the rolling wall’. To-do yang harus dilakukan si robot, yakni bagaimana cara dia harus bisa keluar dari wall yang mengelilingi jalan yang membentuk tiga pola yang sama jenis. Aturan yang ada di challange ini adalah robot harus berputar 3 kali di setiap rolling wall dan robot tidak boleh memakan beacon.

Challenge yang ketiga, yakni si robot harus melewari dua jenis challenge, yakni ‘Moving the beacon and a chosen path’. Pada akhir challange bagian 1, terdapat dua opsi untuk memasuki challenge bagian 2. Ini dapat dihubungkan dengan challenge bagian 1 dengan beberapa cara. Aturan-aturanya sendiri, yakni pada challanfe pertama tidak boleh memakan beacon dan bagaimana caranya beacon hanya perlu dipindahkan saja. Kemudian pada challenge kedua bagaimana caranya ketika robot sudah memilih untuk berada di salah pintu masuk kedua opsi, dia harus berjalan sesuai warna yang nantinya dia masuki. Jika dia memasuki jalan yang berwarna putih maka seterusnya dia juga harus berjalan di jalan yang berwarna putih juga. Bergitupun sebaliknya.

Challenge yang keempat, yakni ‘Pathfinding algorithm’. Ini tentang bagaimana cara si robot melewati jalan yang bercabang.Terdapat beberapa cabang yang setiap cabangnya terdapat beacon. Bagaimana cara si robot dapat menemukan jalan yang kosong(tanpa beacon) dengan beberapa syarat. Syarat challenge ini yang pertama si robot harus mengecek keseluruhan cabang tanpa terkecuali. Kedua, si robot tidak boleh menabrak beacon apalagi memakanya.

Kelima, challenge ini bernama ‘Making random number with repetition’.Challenge ini seperti labirin diawal, tetapi terdapat trik tersembunyi sehingga dapat membentuk sebuah angka. Si robot harus melalui challenge yang terakhir ini dengan beberapa aturan sebelum bertemu dengan beacon-beacon yang ada. Aturanya si robot harus membentuk angka diantara 0-9 secara bebas sebanyak 3 kali dan harus berbeda semua(tidak boleh sama).

Sedikit penjelasan mengenai perantara. Jadi perantara ini adalah beberapa langkah yang digunakan untuk menghubungkan setiap challenge agar tetap berjalan di dalam satu sintaks.

Ini diperlukan karena mempermudah dalam mengolah setiap challenge dan mengurangi resiko gagal dari setiap challenge yang ada. Sebelum garis finish, terdapat pula perantara agar si robot dapat fleksibel dalam memasuki garis finish. Begitulah penjelasan singkat mengenai perantara.

**Explanation of solution for my program problem.**

Baik, saya akan menjelaskan semua solusi dari challenge pada program yang saya buat.

**CHALLENGE PERTAMA**

Pada challenge pertama, yakni ‘simple labirin’. Challenge ini hanya berfokus pada pergerakan si robot untuk bisa menemukan jalan menuju pintu keluar labirin. Sebelum masuk ke pembahasan solusi yang lebih lanjut. Saya akan tampilkan sintaks(solusi) yang saya buat.

repeat(){

if(leftIsObstacle()){

if(frontIsClear()){

forward(1)

}

else{

right()

}

}

else if(~leftisobstacle){

left()

forward(1)

}

if(frontIsBeacon()){

pickUp

west

putdown()

break

}

}

Nah, sintaks diatas merupakan visualisasi untuk menyelesaikan program ini. Saya menggunakan looping-break dan kondisional. Jadi, saya memanfaatkan looping ini untuk moving robot sehingga akan mengurangi manualisasi sintaks(simple dan praktis). Selain itu, looping ini digunakan untuk memastikan bahwa robot akan terus mencari beacon sampai ditemukan. Terdapat pula kondisional yang menjadi bagian looping. Beberapa kegunaan kondisional yang ada dalam looping.

if(leftIsObstacle()){

if(frontIsClear()){

forward(1)

}

else{

right()

}

}

Nah, maksudnya kondisional terebut, yakni robot akan terus bergerak maju sampai menemukan obstacle di sebelah kirinya. Jika ada obstacle di sebelah kiri, robot akan berbelok ke kanan.

Kemudian, pada kondisional dibawah ini maksudnya:

else if(~leftisobstacle){

left()

forward(1)

}

Jika tidak ada obstacle di sebelah kiri, robot akan berbelok ke kiri dan bergerak maju sejauh 1 langkah.

Selanjutnya, kondisional berikut maksudnya:

if(frontIsBeacon()){

pickUp

west

putdown()

break

}

Jika robot menemukan beacon di depannya maka robot akan mengambil beacon, berputar ke arah barat, dan meletakkan beacon.

**CHALLENGE KEDUA**

‘turn on the rolling wall’, Saya menggunakan prosedur, looping, dan kodisional untuk solusi di challenge ini. Jadi pada intinya ada tiga prosedur digunakan untuk mengatur setiap repetisi looping dan penyebutan nama suatu looping yang ada. Karena ada tiga bentuk dari tembok yang menggulung maka diperlukan procedure diketiga bentuk yang ada. Terdapat looping pula diketiga procedure karena aturan menyatakan bahwa harus berputar disetiap gulungan sebanyak tiga kali. Terdapat pula parameter dalam prosedur yang tertera diketiga prosedur tersebut. Terdapat pula sintaks calling procedure yang merupakan pemanggilan dimana masing-masing ada diketiga prosedur. Setiap pemanggilan ini pada masing masing prosedure memiliki parameter yang berisi nilai yang berbeda beda di masing masing procedure. Terdapat pula kondisional dan sekuensial dimasing-masing looping. Satu hal yang saya perlu tekankan bahwa robot akan memindahkan beacon dan meletakanya ditempat yang tidak menghalangi jalanya si looping untuk robot. Ini berdasarkan aturan yang tertera.

Berikut visualisasi codenya:

procedure bagian (n, putaran){

repeat(n){

if (putaran == 1){

north(2)

west(1)

pickup()

west(1)

south(2)

east(2)

right()

putdown()

}

else if(putaran == 2){

north(4)

west(3)

pickup()

left()

left()

putdown()

west(1)

south(4)

east(4)

}

else if (putaran == 3){

north(6)

west(5)

pickup()

west(1)

south(6)

east(6)

north(6)

west(6)

left()

left()

putdown()

}

}

}

# calling prosedure

bagian(1, 1)

bagian(1, 2)

bagian(1, 3)

north(1)

procedure bagian2 (p, muter){

repeat(p){

if (muter == 1){

north(2)

west(1)

pickup()

west(1)

south(2)

east(2)

right()

putdown()

}

else if(muter == 2){

north(4)

west(3)

pickup()

left()

left()

putdown()

west(1)

south(4)

east(4)

}

else if (muter == 3){

north(6)

west(6)

south(5)

pickup

south(1)

east(6)

north(6)

west(6)

left()

left()

putdown()

south(6)

}

}

}

bagian2(1, 1)

bagian2(1, 2)

bagian2(1, 3)

west(1)

procedure bagian3 (p, puter){

repeat(p){

if (puter == 1){

west(2)

south(1)

pickup()

south(1)

east(2)

north(2)

right()

putdown()

}

else if(puter == 2){

west(4)

south(3)

pickup()

left()

left()

putdown()

south(1)

east(4)

north(4)

}

else if (puter == 3){

west(6)

south(5)

pickup()

south(1)

east(6)

left()

left()

putdown()

north(6)

west(6)

south(6)

}

}

}

bagian3(1, 1)

bagian3(1, 2)

bagian3(1, 3)

**CHALLENGE KETIGA**

Pada challenge ini, saya merasa rumit apabila tidak menggunakan bantuan procedure. Oke, saya akan menjelaskan secara garis besar mengenai solusi dalam bentuk sintak yang saya buat. Saya menggunakan 2 prosedure. Prosedure yang bernama curi digunakan untuk menghubungkan bagian rintangan ‘moving the beacon’ ke bagian rintangan berikutnya ‘A chosen path’ pada challenge ini. Lalu pada procedure kedua yang bernama ‘jalan’, saya menggunakan itu untuk pemanggilan opsi salah satu jalur untuk menyelesaikan rintangan bagian pertama sekaligus pada bagian kedua. Karena disitu parameter pemanggilan yang saya gunakan adalah [ jalan(2) ] maka saya menggunakan cara kerja yang kedua yakni melewati beberapa beacon kemudian meneruskan jalur yang berwarna putih. Saya juga dapat menggunakan cara kerja procedur pertama dengan mengganti nilai yang ada dalam parameter tersebut. Terdapat pula looping, conditional, dan variable unutk membantu proses kerja si procedure. Terdapat sintaks pickup dan putdown untuk memenuhi syarat pertama, yakni tidak boleh memakan beacon. Syarat berikutnya juga terpenuhi karena syarat menyatakan bahwa boleh memilih salah satu opsi jalur pada bagian dua. Prosedure [ jalan ] digunakan untuk pemilihan salah satu opsi jalur yang akan dilewati sirobot. Terdapat pula looping dan kondisional dalam menentukan pergerakan robot yang menyesuaikan syarat, yakni ketika jalur putih maka harus pula dengan putih dan sebaliknya.

Berikut visualisasi sintaknya:

n=10

procedure curi{

pickup

north

repeatwhile(frontisclear){

forward

}

right

repeatwhile(frontisclear){

forward

}

right

repeat(n){

forward

}

putdown

left(2)

n=n-1

}

procedure jalan(dipanggil){

if(dipanggil == 1){

repeatwhile(~frontisblack){

if (frontisbeacon){

curi

}

else if(frontisclear){

repeat{

if(frontisblack){

break

}

if(frontisobstacle){

break

}

forward

}

}

else{

left

}

}

repeat{

if(frontisblack&frontisclear){

forward

}

else if (leftisblack&frontisblack){

west

south

break

}

else if(leftisblack&leftisclear){

left

forward

}

else if(rightisblack&rightisclear){

right

forward

}

else if(frontiswhite&leftisclear){

left

forward

}

else if (frontiswhite&rightisclear){

right

forward

}

else{

if(~frontisclear&rightisclear){

right

forward

}

else if(~frontisclear&leftisclear){

left

forward

}

repeatwhile(~frontiswhite&~frontisblack&frontisclear){

forward

}

}

}

}

if(dipanggil == 2){

a = 0

repeatwhile(frontisclear){

forward(1)

if(leftisclear){

left

forward(1)}

if(frontisbeacon){

pickup

forward(1)

left

left

putdown

right

}

a = a + 1

}

P = 1

repeat(){

if(p <= a){

pickup

forward

right

right

putdown

left

left

p = p + 1

if (frontiswhite){

break}

}

}

repeat{

if(frontiswhite&frontisclear){

forward

}

else if(leftiswhite&leftisclear){

left

forward

}

else if(rightiswhite&rightisclear){

right

forward

}

else if(frontisblack&leftisclear){

left

forward

}

else if (frontisblack&rightisclear){

right

forward

}

else if(frontisbeacon&leftisobstacle&~rightiswhite&~rightisblack){

break}

else{

if(~frontisclear&rightisclear){

right

forward

}

else if(~frontisclear&leftisclear){

left

forward

}

repeatwhile(~frontiswhite&~frontisblack&frontisclear){

forward

}

}

}

}

}

jalan(2)

**Challenge keempat**

Solusi challenge ini cukup simple. Saya menggunakan bantuan procedure, looping dan kondisional, serta variable di dalamnya. Prosedure tersebut digunakan untuk mengatur jumlah nilai dalam variable(n) dan juga jumlah nilai dalam looping(x). Terdapat variable untuk menampung suatu nilai yang nantinya digunakan untuk sekuensial. Terdapat looping yang digunakan untuk melakukan perulangan sebuah sekuensial. Syarat pertama telah dipenuhi karena saya menggunakan looping untuk mengecek satu persatu cabang dari jalan yang ada untuk bisa menemukan cabang yang kosong(tidak ada beacon). Syarat kedua juga terpenuhi karena saya menggunakan kondisional agar robot tidak menabrak beacon apalagi memakanya.

Berikut visualisasi sintaksnya:

procedure kotak (n, x){

forward(1)

panjang1 = n

panjang2 = panjang1 \* 2

repeat(x){

if(frontisclear){

north(1)}

if(leftisclear){

west(panjang1)

if(frontisbeacon){

east(panjang2)

}

if(~frontisbeacon){

forward(1)

break

}

west(panjang1)}

}

}

kotak(5, 8)

**CHALLENGE KELIMA**

Ini adala challenge terkahir yang saya buat. Solusi dari challenge ini sebetulnya mudah dan praktis. Lagi- lagi saya menggunakan procedure, variable, looping dan sekuensial dalam menyelesaikanya. Prosedure kotax digunakan untuk mengorganisir perulangan beberapa langkah unutk memtuar(n) dan juga digunakan untuk pemanggilan sebuah kondisi(x). Saya menggunakan variable untuk menentukan langkah langkah sekuensial si robot. Terdapat perantara yang juga merupakan bagian dari challenge ini. Sintaksnya hanya menggunakan looping dan kondisional. Dengan solusi ini, syarat satu yang mentakan bahwa si robot harus membentuk angka dari 0-9 sebanyaa 3 kali dan harus berbeda semua(tidak boleh sama) sebelum bertemu dengan beacon sudah terpenuhi. Saya menggunakan looping dan kondisional dalam membuat tiga angka, yakni 8, 5, dan 9. Pada syarat kedua, kita tidak boleh mengisi langkah si robot dengan angka, tetapi harus menyertakan variabel untuk membantunya.Ini juga terpenuhi karena saya menyatakan setiap gerakan melalui variable untuk menyimpankan sebuah value.

Berikut visualisasi code yang saya buat

procedure kotax (n, x){

kotak0 = 1

kotak1 = kotak0 \* 2

kotak2 = kotak1 \* 2

kotak3 = 3

kotak4 = kotak3 \* 2

kotak5 = kotak2 \* 2

kotak6 = kotak5 \* 2

if(x == 1) {

repeat(n){

paintwhite

north(kotak1)

east(kotak1)

south(kotak1)

west(kotak1)

north(kotak1)

if(frontisobstacle & rightiswhite){

east(kotak1)

stoppainting}

}

}

# west(kotak1)

else if(x == 2){

east(kotak0)

paintwhite

east(kotak3)

north(kotak3)

west(kotak3)

north(kotak3)

east(kotak3)

stoppainting

south(kotak1)

east(kotak0)

}

else if (x == 3){

repeat(n){

paintwhite

east(kotak2)

south(kotak2)

if(rightiswhite){

south(kotak2)

west(kotak2)

east(kotak2)

stoppainting

east(kotak0)

north(kotak4 \* 2)}

if(frontisobstacle & leftisclear){

west(kotak0)

break}

west(kotak2)

north(kotak2)}

}

}

kotax(2, 1)

kotax(0, 2)

kotax(2, 3)

#ini rintangan lanjutan tapi saya masukan sekalian disini karena saya jadikan sebagai perantara

repetisi = 1

if(frontisbeacon){

pickup

left(repetisi \* 2)

putdown

right( repetisi \* 2)

}

repeat(){

if(leftisclear){

left

forward(1)

if(frontisclear){

forward(1)}

}

else if(rightisclear){

right()

forward(1)

}

else if(frontisclear&rightisobstacle&leftisobstacle){

forward(1)

}

if(frontisclear&leftisclear&rightisclear)

{break}

}

#Perantara berkahir