BAB III

ANALISIS MASALAH DAN RANCANGAN PROGRAM

III.1. Analisis Masalah

Permainan *Halma* merupakan permainan yang mengasah logika pemainnya. Permainan *halma* mengharuskan pemainnya untuk memindahkan pion-pion yang dimiliki setiap pemain ke posisi tujuannya secepat mungkin. Permasalahan yang dihadapi bagaimana sistem dapat mencari solusi penyelesaian dengan membentuk langkah-langkah yang didapat pada masing-masing kondisi dan memindahkan pion-pion tersebut secepat mungkin untuk memenangkan pertandingan.

Sistem yang dibangun untuk pencarian solusi menggunakan pendekatan metode *Deep First Search* (DFS). Pencarian dimulai dari level paling pertama (level 0) kemudian dilanjutkan ke anak paling kiri pada level berikutnya (level 1) demikian seterusnya sampai tidak terdapat anak lagi.

Hasil yang dicapai dari implementasi sistem ini adalah menemukan kemungkinan solusi sehingga menghasilkan langkah yang optimal. Sehingga sistem mampu memindahkan seluruh pion yang dimiliki secepat mungkin dengan langkah yang optimal.

III.2. Aturan Permainan

Cara permainan *halma* adalah sebagai berikut :

1. Mulai melangkahkan pion dengan arah yang diizinkan secara bergiliran.
2. Bidak hanya melangkah satu langkah apabila didepannya area kosong. Namun tidak dapat melangkah lebih dari satu langkah dengan meloncati bidak-bidak lain, dengan tetap mengikuti alur garis yang ada.
3. Dalam melangkah, masukkan bidak ke area segitiga seberang yang sama warnanya dan susun secara tepat agar area segitiga dapat terisi dengan penuh.
4. Pemenangnya adalah pemain yang paling cepat memindahkan semua bidaknya ke area segitiga di seberang.

III.3. Perancangan

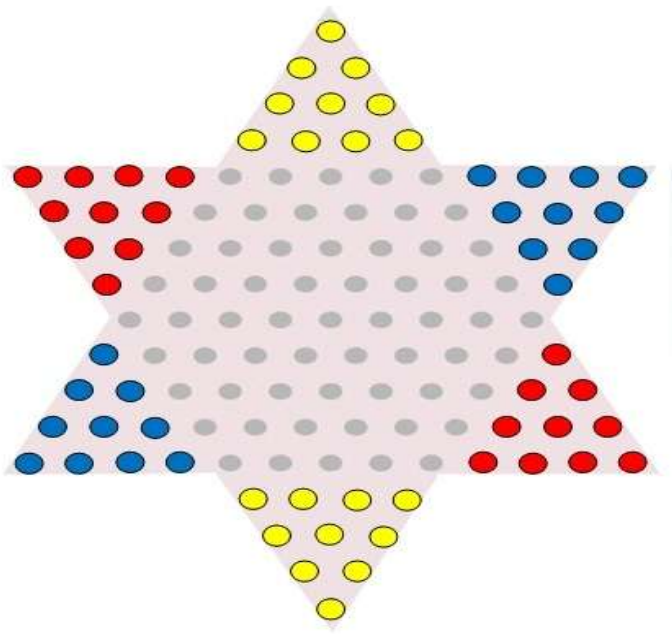
Perancangan bertujuan untuk memenuhi kebutuhan penggunaan aplikasi dan memberikan gambaran mengenai program kepada pengguna aplikasi. Kegiatan yang dilakukan pada tahap ini adalah perancangan dari semua sistem aplikasi. Tahap ini merupakan tahap yang menentukan keberhasilan dari perancangan program.

III.3.1. Proses Perancangan Gambar Papan *Halma*

Papan permainan *Halma* dirancang dengan menggunakan aplikasi *Microsoft Visio* dan kemudian di-*copy and paste* ke aplikasi *Adobe Photoshop CS3* dan disimpan.

Papan permainan *halma* dirancang dengan menggunakan *Ellipse tool* untuk menggambar lingkaran (bulatan) kecil dan dilakukan proses *fill color* dengan warna hitam untuk menghasilkan lingkaran hitam kecil. Sedangkan garis-garis pada papan permainan dirancang dengan menggunakan *Line tool*. Proses terakhir, dirancang tiga buah segitiga sama sisi dengan cara menggambarkan garis-garis yang berhubungan secara berturut-turut hingga membentuk sebuah segitiga dan dilakukan proses *fill color* dengan warna kuning, merah dan biru.

Gambar yang dihasilkan tersebut di-*copy and paste* ke aplikasi *Adobe Photoshop CS3* untuk disimpan ke dalam format gambar *.GIF



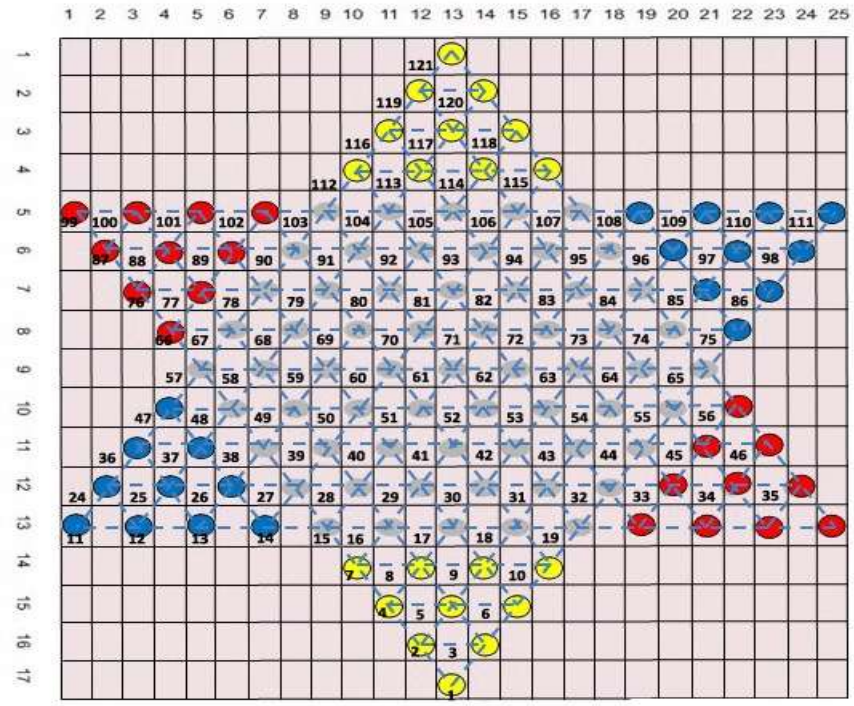
Gambar III.1 Rancangan Papan *Halma*

III.3.2. Proses Inisialisasi Gambar Papan *Halma*

Rancangan papan *halma* diberi inisialisasi. Pada **gambar III.2** memperlihatkan bahwa terdapat matrik posisi berukuran 17 x 25 dimulai dari posisi [1,1] sampai posisi [17,25]. Papan *halma* yang diberi tanda bulat disebut *node* yang jumlahnya 121 buah. Setiap *node* terletak pada sebuah matrik posisi. Pada *node* 1 berada pada matrik [17,13] dan *node* 61 berada pada matrik [9,13]. Perhatikan *node* 70 dimana memiliki tetangga *node* 80, 81, 69,71, 60, 61. Jadi setiap *node* masing-masing memiliki tetangga. Sebuah *node* terdapat maksimum 6 buah *node* tetangga. Namun terdapat *node* yang tetangganya lebih kecil dari 6 *node* tetangga artinya ada *node* tetangga tidak berada di area papan *halma* diberi nilai 0.

Semua *node* diberi nilai sesuai dengan gambar diatas. Masing-masing *node* terdapat enam buah pointer yaitu :

1. S [1] berisi nilai label dari posisi yang terletak di samping kiri *node*.
2. S [2] berisi nilai label dari posisi yang terletak di samping kanan *node*.
3. A [1] berisi nilai label dari posisi yang terletak di kiri atas *node*.
4. A [2] berisi nilai label dari posisi yang terletak di kanan atas *node*.
5. B [1] berisi nilai label dari posisi yang terletak di kiri bawah *node*.
6. B [2] berisi nilai label dari posisi yang terletak di kanan bawah *node*.



Gambar III.2. Inisialisasi Papan Permainan *Halma*

Untuk lebih jelasnya dilihat pada contoh di bawah ini :

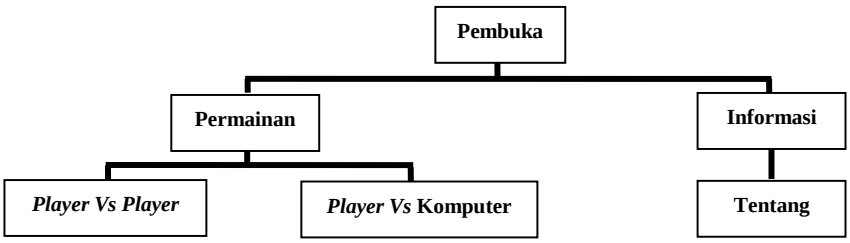
Node (1) diposisi [13,17] memiliki nilai pointer S [1] = 0, S [2] = 0, A [1] = 2, A [2] = 3, B [1] = 0, B [2] = 0.

Node (5) diposisi [13,15] memiliki nilai pointer S [1] = 4, S [2] = 6, A [1] = 8, A [2] = 9, B [1] = 2, B [2] = 3.

III.3.3. Diagram Hirarki

Perancangan diagram hirarki atau diagram bertingkat dimaksudkan untuk memberikan gambaran visual sehingga mempermudah proses perancangan program aplikasi. Diagram ini dibuat agar pengguna aplikasi lebih memahami

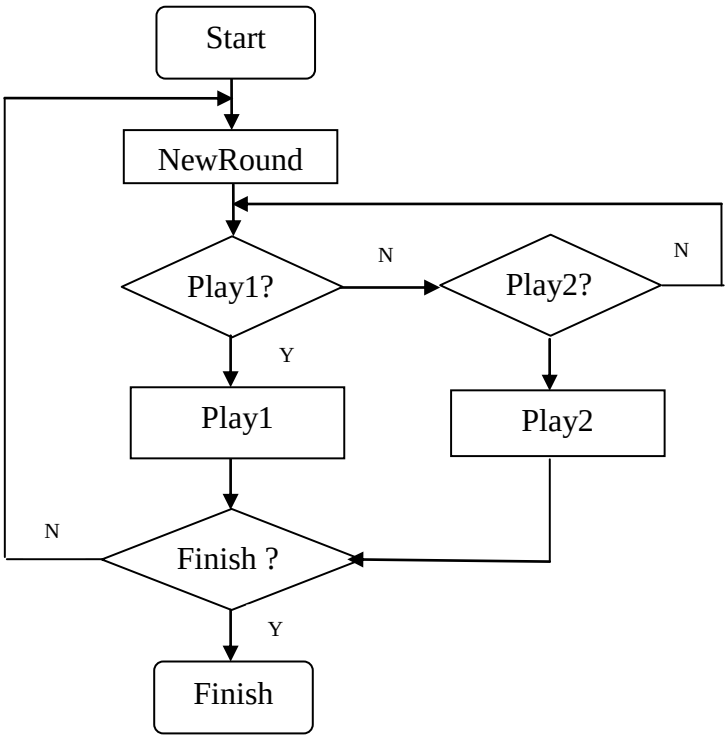
struktur-struktur yang ada dalam aplikasi. Dalam diagram hirarki ini akan diberikan gambaran mengenai tingkat menu pada aplikasi.



Gambar III.3. Diagram Hirarki Permainan *Halma*

III.3.4. Diagram *Flowchart*

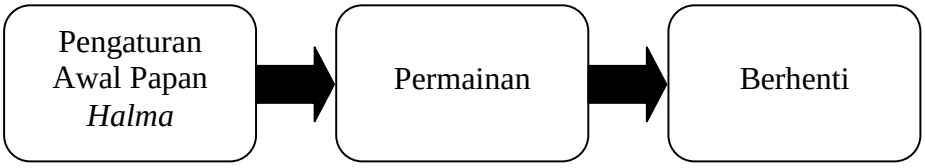
Diagram *Flowchart* perlu dibuat sebelum melakukan proses pemograman agar dapat bekerja sebagai alat ukur bagaimana perangkat lunak ini dirancang. Diagram *Flowchart* merupakan alur sistem permainan yakni, pemain bisa memilih antara bermain dengan pemain lain atau bermain dengan komputer. Pemain akan memulai dengan ronde baru yakni papan *halma* dalam kondisi awal. Setiap pemain mendapat giliran secara berurutan dan setiap pemain akan mendapatkan tampilan langkah-langkah yang dapat diambil oleh pemain dan langkah terpendek yang ditempuh untuk mencapai tujuan. Setelah permainan selesai, sistem akan memberitahukan bahwa pemain mana yang tiba sampai tujuan atau yang memenangkan *game halma* tersebut



Gambar III.4. *Flowchart* Permainan *Halma*

III.3.5. Perancangan Proses Permainan

Pada perancangan perangkat lunak permainan *halma* ini, terdiri dari beberapa proses yaitu proses pengaturan awal papan *halma*, proses pengecekan langkah yang dapat dijalankan oleh pemain, proses pencarian langkah terpendek, proses pengecekan pemenang dan berhenti.



Gambar III.5. Blok Diagram Proses Perancangan

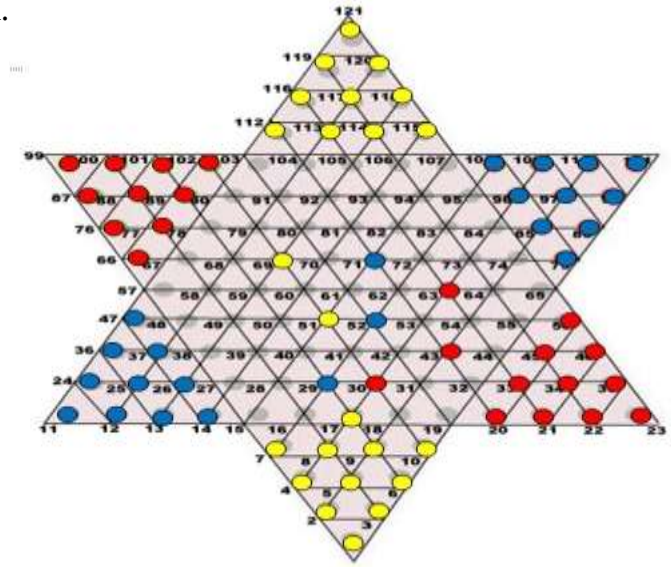
III.3.6. Proses Pengecekan Langkah-Langkah Yang Dapat Dijalankan Oleh Biji

Setiap pemain dapat menggerakkan bijinya ke posisi yang diinginkan. Namun posisi yang diinginkan tersebut harus dapat dijalankan. Jika tidak, maka pergerakan biji tidak diperbolehkan. Proses pengecekan pergerakan biji yang diperbolehkan adalah sebagai berikut :

1. Pengecekan dimulai dari posisi awal biji dengan mengecek nilai setiap pointer dari posisi biji tersebut.
2. Nilai pointer menunjukkan posisi tujuan yang dapat digerakkan oleh biji tersebut. Jika nilai pointer bernilai 0, maka berarti biji tidak dapat digerakkan ke arah tersebut.
3. Jika posisi tujuan yang dapat digerakkan masih kosong (tidak ditempati oleh biji), maka biji berhenti di posisi tersebut dan tidak dapat digerakkan lagi.
4. Jika posisi tujuan yang dapat digerakkan tidak kosong (ditempati oleh biji), maka biji tersebut digerakkan ke posisi dengan arah pointer yang sesuai dengan arah posisi tujuan tersebut jika ditinjau sebagai nilai pointer dari posisi asal.
5. Proses pengecekan untuk langkah ke empat dilakukan untuk semua nilai pointer dari posisi tujuan tersebut yang telah ditempati oleh biji hingga tidak terdapat nilai pointer dari posisi tujuan yang telah ditempati oleh biji.

6. Jika pada waktu proses pengecekan, didapat posisi tujuan yang telah diperoleh sebelumnya, maka proses pengecekan untuk posisi tujuan tersebut tidak perlu dilanjutkan lagi.

Agar lebih jelas, berikut contoh proses ditengah-tengah permainan. Misalkan ingin digerakkan biji kuning yang berada pada posisi 17 dengan posisi biji-biji lainnya.



Gambar III.6. Contoh Keadaan Posisi Biji-Biji Pada Permainan *Halma*

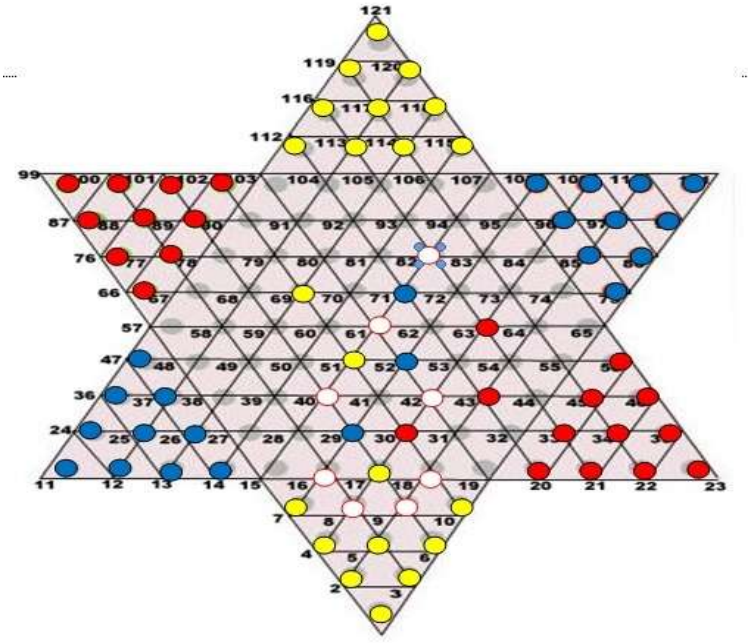
Posisi tujuan yang dapat dicapai oleh biji kuning pada posisi 17 tersebut adalah sebagai berikut :

1. Posisi 8 yang merupakan nilai pointer B1 dari posisi 17.
2. Posisi 9 yang merupakan nilai pointer B2 dari posisi 17.
3. Posisi 16 yang merupakan nilai pointer S1 dari posisi 17.
4. Posisi 18 yang merupakan nilai pointer S2 dari posisi 17.
5. Nilai pointer A1 dari posisi 17 yaitu posisi 29 telah ditempati oleh biji, maka posisi tujuan yang dapat ditempati oleh biji adalah sesuai dengan nilai pointer A1 dari posisi 29 yaitu **posisi 40**.

6. Karena posisi 40 ditempati dengan melompati biji lainnya, maka biji masih dapat digerakkan lagi, namun harus dengan melakukan lompatan, yang berarti bahwa posisi tujuan sesuai dengan nilai pointer dari posisi tersebut harus telah ditempati oleh biji lainnya. Jika tidak, maka biji tidak dapat digerakkan lagi. Nilai pointer dari posisi 40 hanya pointer A2 yaitu posisi 51 dan B2 yaitu posisi 29 yang telah terisi, maka pointer A2 dari posisi 51 yaitu **posisi 61**. Pengecekan dilakukan untuk posisi 61. Nilai pointer dari posisi 61 hanya pointer A2 yaitu posisi 71 dan pointer B2 yaitu posisi 52 yang telah ditempati oleh biji maka pointer A2 dari posisi 71 yaitu **posisi 82** dan pointer B2 dari posisi 52 yaitu **posisi 42** merupakan posisi tujuan yang dapat ditempati oleh biji.
7. Pengecekan dilanjutkan untuk posisi 82 dan posisi 42. Nilai pointer dari posisi 82 hanya pointer B1 yaitu posisi 71 yang telah ditempati oleh biji, maka pointer B1 dari posisi 71 yaitu posisi 61 merupakan posisi tujuan yang dapat ditempati oleh biji. Namun, karena posisi 61 merupakan posisi asal sebelumnya dan telah dimasukkan sebagai posisi tujuan, maka proses pengecekan untuk posisi 61 dihentikan. Nilai pointer dari posisi 42 hanya pointer A1 yaitu posisi 52, pointer B1 yaitu posisi 30, dan pointer S2 yaitu posisi 43 yaitu posisi 44 merupakan posisi tujuan yang dapat ditempati oleh biji. Namun, karena posisi 17 merupakan posisi awal maka posisi 17 bukan merupakan posisi tujuan **posisi 44**.
8. Nilai pointer A2 dari posisi 17 yaitu posisi 30 telah ditempati oleh biji, maka pointer A2 dari posisi 30 yaitu posisi 42 merupakan posisi tujuan yang dapat

ditempati oleh biji. Namun, karena posisi 42 sebelumnya telah dimasukkan sebagai posisi tujuan, maka proses pengecekan untuk posisi 42 dihentikan.

Maka posisi tujuan yang dapat dicapai oleh posisi 17 adalah posisi 8, 9, 16, 18, 40, 42, 44, 61, dan 82

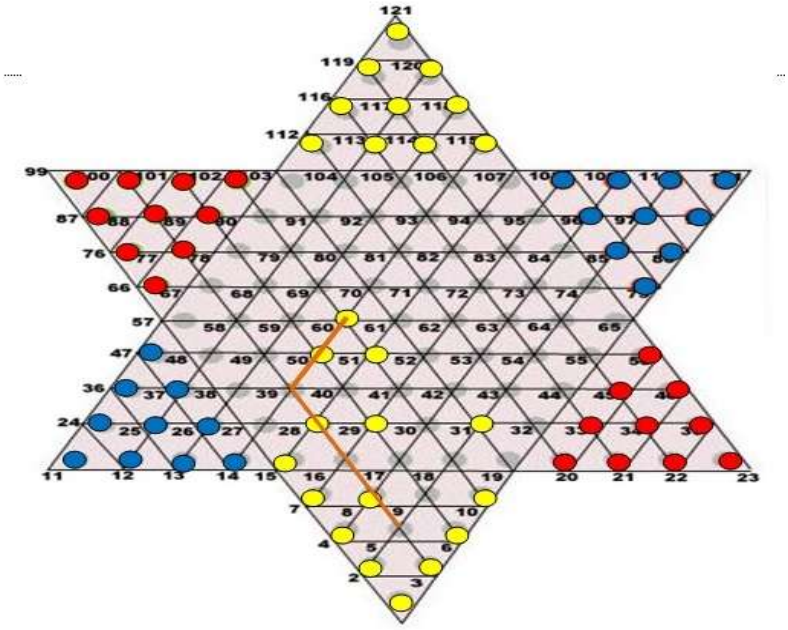


Gambar III.7. Contoh posisi tujuan dari biji pada permainan *halma*

III.3.7. Proses Pencarian Langkah Terpendek

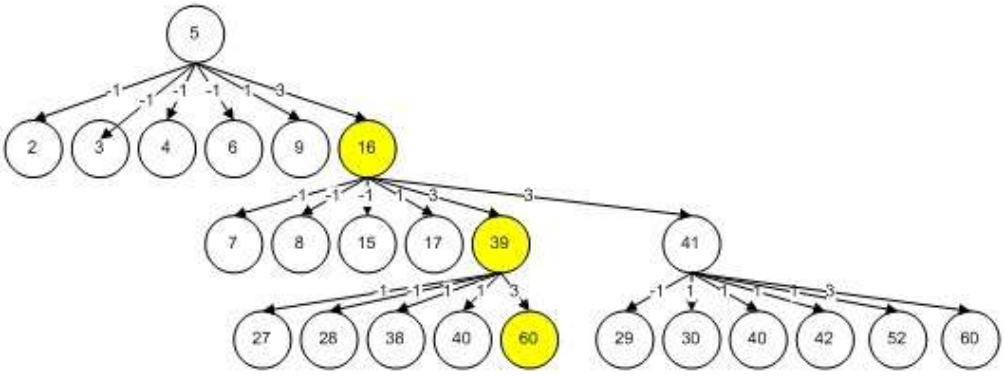
Pada proses ini kita menggunakan metode *Depth First Search* (DFS), proses akan dilakukan pada semua anaknya sebelum dilakukan pencarian ke node-node (titik) yang selevel. Pencarian dimulai dari *node* akar ke level yang lebih tinggi.

Setiap langkah diberi bobot nilai untuk mengecek pencarian jalur dengan menggunakan DFS, yakni mengambil jalur sebelah kiri terlebih dahulu



Gambar III.8. Contoh Kondisi Penerapan DFS

Proses pencarian langkah terpendek dari awal dengan menelusuri ke dalam sejauh mungkin sebelum kembali ke pencarian awal



Gambar III.9. Pohon Ruang Solusi

Dimana bobot nilai ditentukan :

-1 : Terdapat biji *halma*, jadi langkah tidak dapat dilakukan.

1 : Daerah kosong, namun hanya dapat berhenti di titik tersebut dan tidak dapat melakukan lompatan lagi.

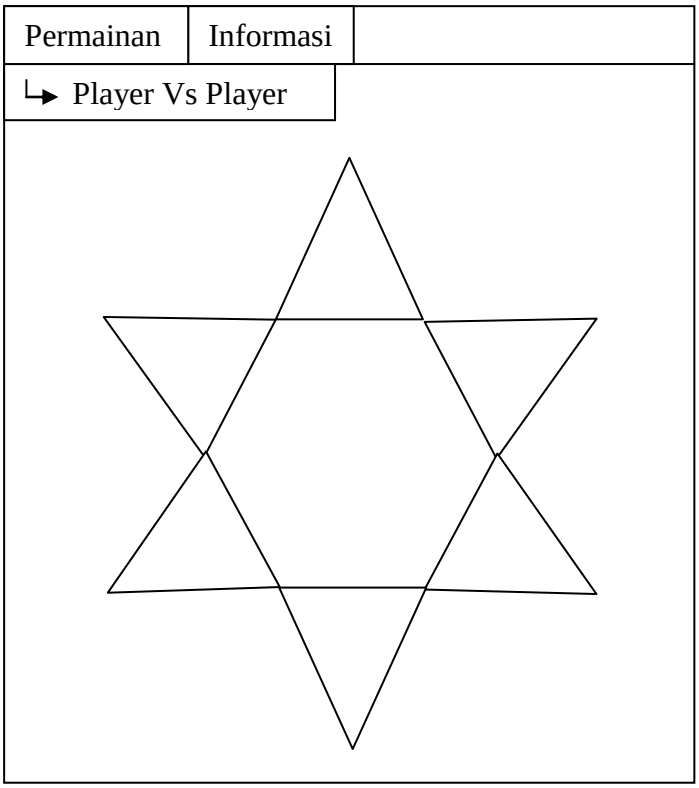
3 : Daerah kosong yang dapat dicapai dengan melalui sebuah lompatan dari biji *halma* lain.

III.3.8. Proses Pengecekan Pemenang

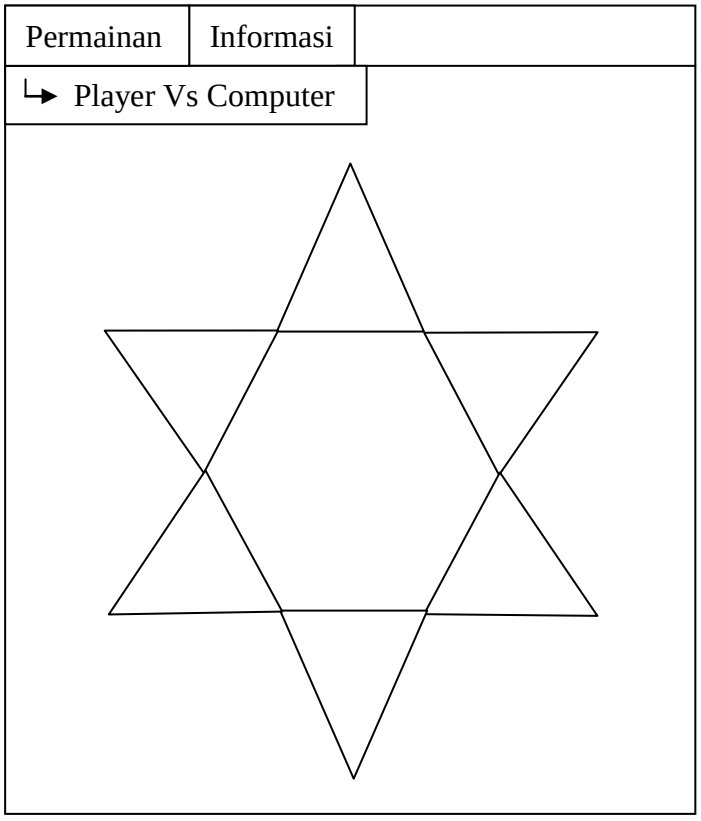
Pada proses ini akan dilakukan pengecekan terhadap biji-biji yang telah masuk ke daerah tujuan rumah apakah semuanya sudah masuk atau tidak dengan cara menyimpan *array* posisi tujuan rumah. Pemain yang duluan memasukkan semua bijinya ke daerah tujuan rumah dinyatakan sebagai pemenang.

III.3.9. Perancangan Layar

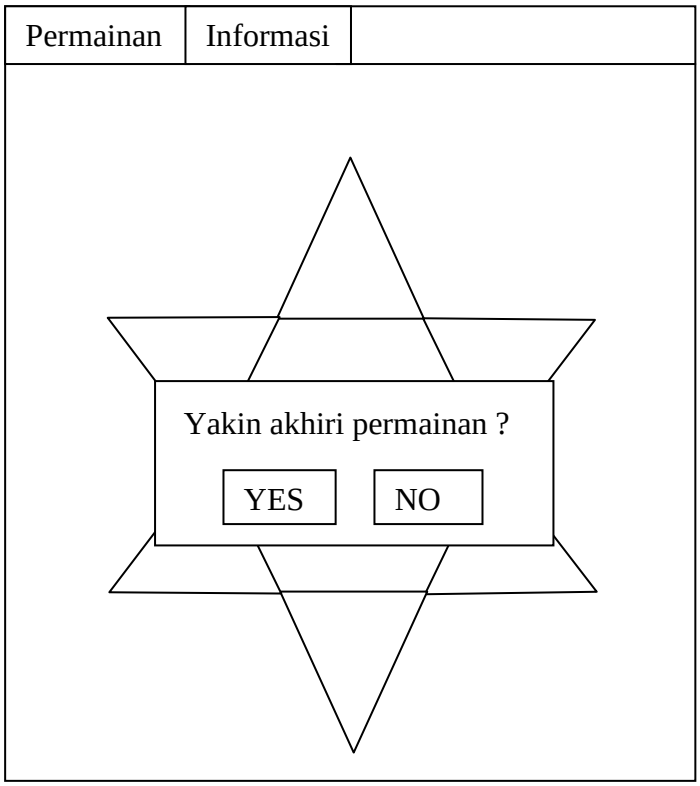
Pada perancangan layar permainan *halma*, terdiri dari beberapa modul yaitu modul permainan dan modul informasi. Modul permainan dibagi lagi menjadi beberapa menu yaitu menu *player versus player*, *player versus* komputer dan berhenti. Sedangkan pada modul informasi hanya terdapat menu tentang, yaitu menu pemberitahuan permainan *game halma* tersebut.



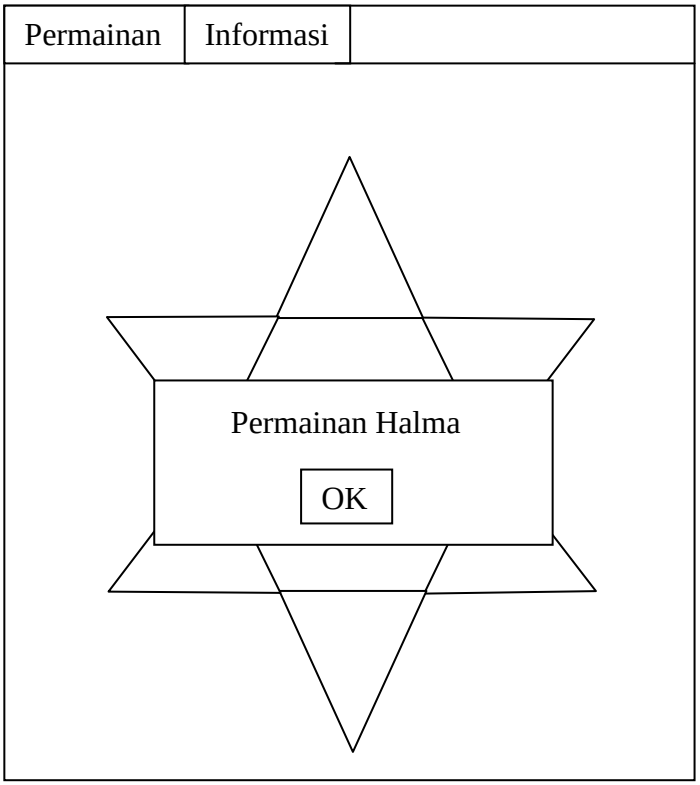
Gambar III.10. Tampilan *Player Versus Player*



Gambar III.11. Tampilan *Player Versus* Komputer



Gambar III.12. Tampilan Menu Berhenti



Gambar III.13. Tampilan Menu Informasi

III.4. Pembuatan

Setelah melakukan perancangan, langkah selanjutnya adalah proses pembuatan. Pembuatan permainan *game halma* merupakan realisasi dengan rancangan program yang telah dilakukan pada tahap perancangan sebelumnya. Langkah-langkah pembuatan permainan *game halma* ini adalah sebagai berikut :

1. Pembuatan *user interface* dan pengisian *coding*.
2. Pengecekan *user interface* beserta modul-modulnya.
3. Pengujian proses permainan.

III.4.1. Perangkat Pembuatan

Pembuatan permainan *game halma* menggunakan 1 buah komputer dan 1 buah laptop. Perangkat keras laptop sebagai berikut :

1. Komputer dengan *processor Intel Pentium Core 2 Duo 2.2 Ghz*.
2. *RAM* berukuran 1 Gb.
3. *Hardisk* berukuran 320 Gb.
4. LCD.
5. *Mouse* dan *keyboard*.

Perangkat keras komputer adalah sebagai berikut :

1. Komputer dengan *processor Intel Pentium IV*
2. *RAM* berukuran 1 Gb.
3. *Hardisk* berukuran 80 Gb.
4. LCD.
5. *Mouse* dan *keyboard*.

Perangkat lunak yang digunakan terdiri dari :

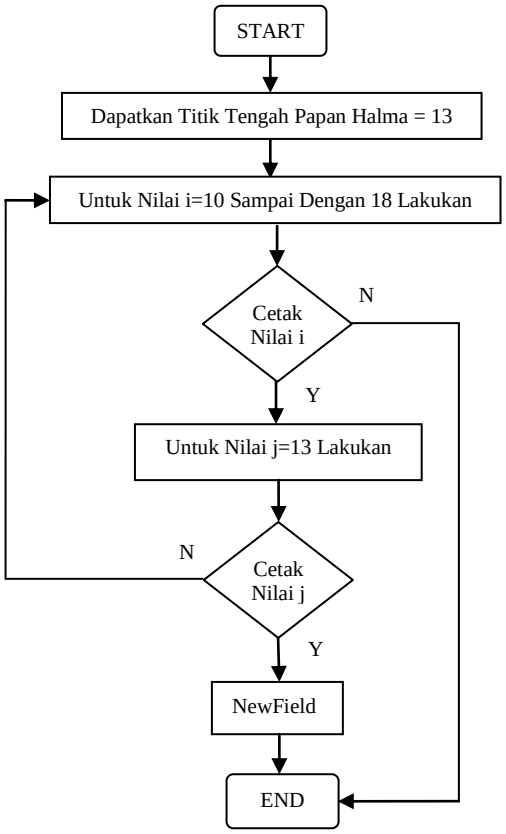
1. *Netbeans IDE 7.0*
2. Sistem operasi *Windows Seven*

III.4.2. Pengkodean

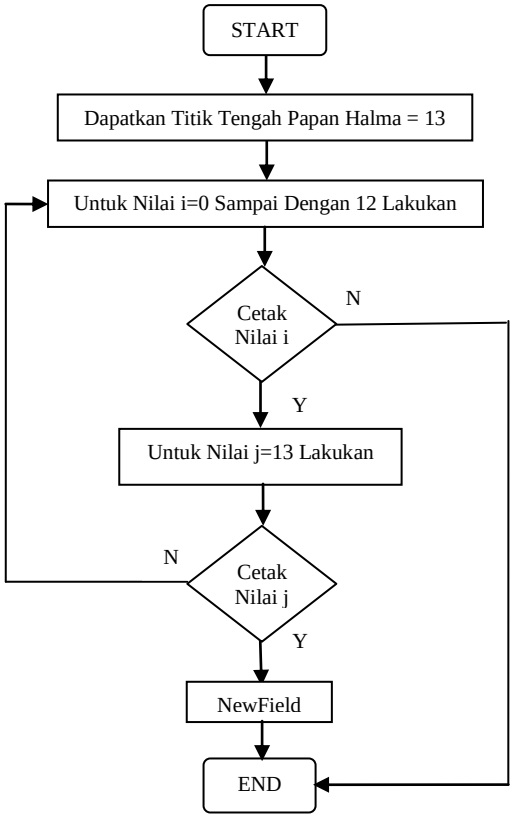
Bagian aplikasi permainan *halma* dikodekan dengan menggunakan bahasa pemograman *Netbeans IDE 7.0.*

III.4.2.1. Algoritma Pengesetan Board *Halma*

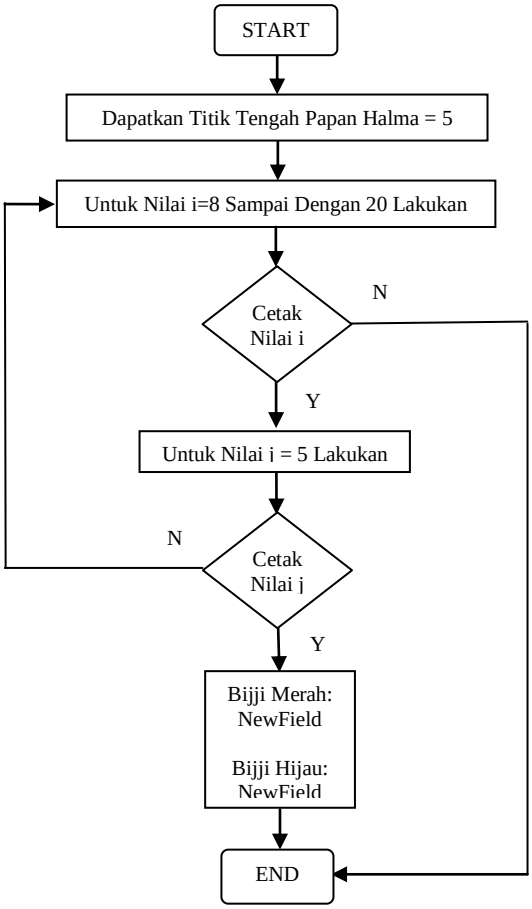
Mengisi titik-titik kosong untuk permainan ditengah-tengah papan *halma*



Membuat Papan *Halma* berbentuk bintang



Membangun Posisi Masing-masing Biji *Halma* , Merah dan Hijau



III.4.2.2. Algoritma Pencarian Langkah