**LAPORAN TUGAS BESAR**

**KKBTI4014 – Dasar-dasar Pemrograman**

**ULAR TANGGA**



**Disusun oleh**

**HILDA ANISSA NUR MAULUDINE – 141524009**

**WAFFI FATUR RAHMAN – 141524030**

**Program Studi DIV Teknik Informatika**

**Departemen Teknik Komputer dan Informatika**

**Politeknik Negeri Bandung**

**2014**

1. **DESKRIPSI UMUM**

Sejarah permainan ular tangga itu menurut sumber yang dibaca permainan ular tangga ditemukan oleh orang india sejak dahulu kala. Permainan ular tangga adalah permainan anak-anak yang biasa dilakukan oleh 2 orang atau lebih. Permainan ular tangga ini dimainkan di atas sebuah papan yang di dalamnya ada kotak kotak kecil, dalam beberapa kotak tersebut terdapat ular dan tangga yang saling terhubung dengan kotak-kotak lainnya. Bagian-bagian yang ada dalam permainan ini terdiri dari :

* Menu untuk menentukan tipe permainan
* level untuk menentukan level permainan
* player untuk menentukan jumlah pemain
* home untuk lokasi awal bidak
* bidak untuk menentukan bentuk pemain
* turn player untuk menunjukan giliran pemainan
* player position untuk menyatakan posisi pemain
* memulai permainan

Permainan di mulai dengan posisi awal pada kotak pertama lalu menentukan urutan bermain dan menentukan bidak masing-masing.

* proses permainan

Setiap pemain melempar dadu sesuai dengan urutannya. Dadu yang muncul menunjukkan jumlah langkah yang akan dilakukan. Jika dadu muncul angka 6 maka pemain tersebut mendapat giliran satu kali. Jika bidak menempati lokasi tangga atau ular maka bidak akan berpindah sesuai lokasi yang telah ditentukan sebelumnya. Jika pemain menempati lokasi yang mana terdapat pemain lainnya maka pemain yang sebelumnnya berasa di lokasi tersebut akan di gantikan dan kembali ke home.

* akhir permainan

permainan berakhir bila salah satu pemain mencapai petak terakhir.

Variasi yang ada pada permainan ular tangga yang berkembang di masyarakat tidak ada yang permanen maksudnya, tidak ada papan permainan standar dalam ular tangga ini. Maka dari itu, setiap orang dapat menciptakan papan mereka sendiri dengan jumlah kotak, ular dan tangga yang berlainan.

1. **DESKRIPSI APLIKASI**

Permainan ular tangga yang kami buat sama halnya dengan permainan ular tangga pada umumnya, pada permainan ular tangga ini memiliki papan permainan berukuran 10x10 jadi secara keseluruhan memiliki 100 array, pada beberapa array disisipkan ular atau tangga yang ditentukan secara random, apabila pemain berada pada array berisi tangga maka dengan otomatis pemain akan menuju array yang telah ditentukan yang lebih besar dari element array tersebut begitu pula sebaliknya jika pemain berada pada array berisi ular maka pemain akan menginjak bidak permainan dengan element yang lebih rendah dari sebelumnya. Permainan ular tangga ini di lakukan oleh 2 hingga 4 orang, maka jika 2 pemain berada pada array dengan nomor yang sama, pemain awal yang sudah menempati array tersebut harus mengulangi permainan mulai dari array pertama lagi. Jika pemain mengeluarkan angka 6 pada kocokan dadu, maka dengan otomatis pemain tersebut mendapatkan kesempatan untuk mengocok dadunya kembali.

Tampilan pertama yang kami buat pada saat pemain ingin memulai permainan ialah:

1. Menu game

Pada tampilan menu game ini, kami menyediakan pilihan yang akan pemain pilih, pilihannya antara lain:

1. Single player

Pada menu pilihan single player ini kami membuat permainan untuk 1 orang pemain dan 3 pemain lainnya adalah komputer.

1. Multi player

Pada menu pilihan Multi Player ini kami membuat permainan untuk 2 hingga 4 pemain.

1. Help

Pada menu pilihan Help, kami membuat sebuah petunjuk bagaimana caranya memainkan permainan ular tangga tersebut.

1. Exit

Menu Exit ini cara untuk keluar dari permainan

1. Level

Setelah pemain memilih menu game, selanjutnya pemain akan memilih level sesuai tingkat kesulitan yang diinginkan, terdapat 4 level yaitu:

1. Easy

Pada level easy yang dipilih pemain, kami menyediakan permainan paling mudah dengan menggunakan 7 tangga dan 3 ular yang tersebar secara random pada setiap papan permainan tersebut.

1. Medium

Pada level medium yang dipilih pemain, kami menyediakan 5 tangga dan 5 ular yang tersebar secara random pada papan permainan tersebut.

1. Hard

Level hard, sesuai namanya level ini kami sediakan dengan level permainan yang bisa dibilang sulit, dengan menyisipkan 7 ular dan hanya 3 tangga.

1. Lucky

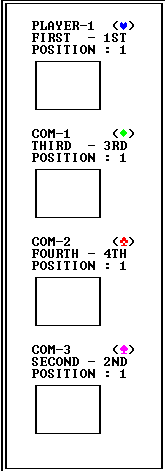
Level lucky ini dibuat sebagai variasi pada permainan ular tangga yang dibuat, level lucky ini kami menyisipkan 7 ular dan 7 tangga, dan yang bisa menyelesaikan permainan ini hanyalah orang-orang yang lucky atau beruntung.

1. Symbol

Untuk menandai setiap pemain kami menggunakan symbol symbol yang biasa ada pada kartu tarot dan juga berwarna agar dengan mudah dapat membedakan symbol yang satu dengan yang lain, symbol-symbol tersebut, yaitu:

1. Heart, berwarna biru
2. Diamond, berwarna hijau
3. Club, berwarna merah
4. Spade, berwarna ungu

Selanjutnya setelah memilih opsi-opsi diatas, pemain akan melihat tampilan papan permainan.

Pada bagian paling kiri pada papan permainan terdapat tabel seperti disamping ini yang berisi profil pemain, urutan pemain, menampilkan posisi dimana pemain sedang singgah dan juga menampilkan tampilan kocakan dadu.

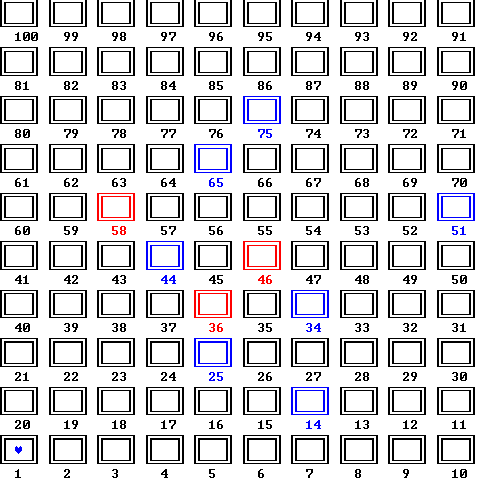
Profil pemain

Urutan pemain

Posisi pemain

Dadu pemain

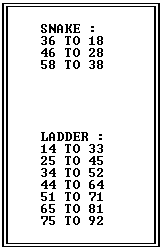
Selanjutnya pada tampilan bagian tengah papan permainan terdapat bidak bidak yang telah berisi ular dan tangga sesuai level yang di pilih.



Tangga

Ular

Pada bagian kanan atas pada tampilan papan permainan yang dibuat, terdapat tampilan untuk menentukan pasangan bidak ular dan tangga yang dirandom, misalnya bidak nomor 36 berisi ular berpasangan dengan bidak nomor 18 yang nantinya jika pemain menginjak angka 36 maka otomatis akan berpindah bidaknya menuju bidak nomor 18 yang angkanya lebih rendah.



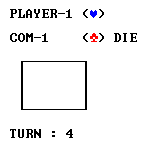
Pada bagian bawah kanan pada papan permainan terdapat tampilan untuk mengocok dadu, pengocokan dadu dilakukan secara random oleh komputer. Pada tampilan ini pemain juga mendapatkan petunjuk siapa pemain yang sedang bermain sesuai gilirannya masing-masing dan menujukan perolehan dadu yang telah dikocok, cara mengocok dadu sangatlah mudah, pemain dapat menekan tombol apapun pada keyboard komputernya.

 Menunjukan pemain yang sedang aktif.

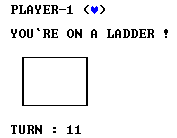
Menunjukan perintah untuk pemain.

Menunjukan berapa dadu yang didapat

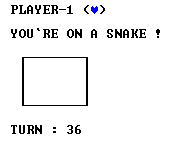
Menunjukan banyak putaran yang terjadi



Menunjukan bahwa pemain harus kembali ke start awal karena bidak yang sedang ditampati telah ditempati juga oleh pemain lain.



Menunjukan kondisi dimana pemain telah berada pada posisi bidak berisi tangga



Menunjukan kondisi dimana pemain sedang berada pada posisi bidak berisi ular.

**III. DESKRIPSI KEBUTUHAN DATA**

1. Data pemain

type theplayer = record

turn : integer //menunjukkan giliran pemain bersangkutan

symbol : integer //menyimpan karakter pemain mainkan

position : integer //menyimpan lokasi keberadaan pemain

End

1. Data jumah pemain dan jumlah komputer

player\_com : array [1..4] of theplayer // menampung maksimal 4 pemain

player : integer // jumlah pemain (bukan komputer)

computer : boolean // penentu ada atau tidak ada komputer

jika ada maka jumlah komputer = 4 – player

jika tidak maka jumlah komputer = 0

all\_player : integer //jumlah pemain + jumlah komputer

1. Data lokasi serta banyaknya ular dan tangga

snake : integer //menyimpan banyaknya ular

ladder : integer //menyimpan banyaknya tangga

snakefield : array [1..7][1..2] of integer //menyimpan banyaknya ular maksimal 7

menyimpan posisi ekor dan kepala ular

ladderfield : array [1..7][1..2] of integer //menyimpan banyaknya tangga maksimal 7

menyimpan posisi awal dan akhir tangga

1. Data papan permainan

field : array [1..100] of character //menyimpan letak pemain pada papan

1. Data jumlah putaran permainan

Count\_turn : integer //menyimpan jumlah putaran yang terjadi

1. Data lainnya

turn : integer //menunjukkan pemain yang aktif

winner : integer //menunjukkan pemenang permainan

dice : integer //menyimpan nilai dadu yang keluar

keterangan : boolean //membantu pembentukan kondisi

i,j,k,l : integer //membantu pembentukan loop

1. **RANCANGAN PROSES**

**main**

**exit**

**write\_score**

**opening**

**menu\_game**

**play\_game**

**single\_player**

**multi\_player**

**help**

**show\_score**

**level**

**player\_number**

**computer\_option**

**symbol**

**field\_show**

**random\_ular\_tangga**

**random\_turn**

**random\_symbol**

**profil**

**roll\_dice**

**gotoxy**

**cursor**

|  |  |  |
| --- | --- | --- |
| Nama Modul | : | Gotoxy |
| Author / penanggung jawab | : | Anonymous |
| Jenis | : | Prosedur |
| I.S | : | Kursor pada posisi akhir output |
| F.S | : | Kursor pada posisi (x,y) |
| **Algoritma**  void gotoxy(int x, int y)  {  COORD coord;  coord.X = x;  coord.Y = y;  SetConsoleCursorPosition(GetStdHandle(STD\_OUTPUT\_HANDLE), coord);  **}** | | |

|  |  |  |
| --- | --- | --- |
| Nama Modul | : | Cursor |
| Author / penanggung jawab | : | Waffi Fatur Rahman |
| Jenis | : | Fungsi |
| Input | : | Jumlah pilihan/menu yang tersedia  Posisi awal kursor (x,y)  Keyboard ( atas / bawah / kiri / kanan ) |
| Output | : | Angka yang mewakili pilihan/menu yang dipilih |
| **Algoritma**  int cursor(int option,int x,int y)  {  int select=1;  int input;  gotoxy(x,y);printf("%c",16);  do  {  input=getch();  if(input==80)  {  gotoxy(x,y);printf(" ");  y++;select++;  if(select>option)  {  y=y-option;  select=1;  }  gotoxy(x,y);printf("%c",16);  }  if(input==72)  {  gotoxy(x,y);printf(" ");  y--;select--;  if(select<1)  {  y=y+option;  select=option;  }  gotoxy(x,y);printf("%c",16);  }  }  while(input!=13);  return select;  } | | |

|  |  |  |
| --- | --- | --- |
| Nama Modul | : | Opening |
| Author / penanggung jawab | : | Hilda Anissa Nur Mauludine |
| Jenis | : | Prosedur |
| I.S | : | Layar kosong |
| F.S | : | Tampil animasi pembukaan game |
| **Algoritma** | | |

|  |  |  |
| --- | --- | --- |
| Nama Modul | : | menu\_game |
| Author / penanggung jawab | : | Hilda Anissa Nur Mauludine |
| Jenis | : | Fungsi |
| Input | : | Nilai 1 s.d 5 ( menggunakan modulcursor ) |
| Output | : | Nilai 1 s.d 5 |
| **Algoritma**  int menu\_game()  {  return cursor(4,52,26);  } | | |

|  |  |  |
| --- | --- | --- |
| Nama Modul | : | single\_player |
| Author / penanggung jawab | : | Hilda Anissa Nur Mauludine |
| Jenis | : | Prosedur |
| I.S | : | Jumlah pemain dan komputer belum diassign |
| F.S | : | Jumlah pemain = 1  Komputer diaktifkan  Memanggil modul level dan modul symbol. |
| **Algoritma**  void single\_player()  {  level();  player=1;  computer=true;  symbol();  } | | |

|  |  |  |
| --- | --- | --- |
| Nama Modul | : | multi\_player |
| Author / penanggung jawab | : | Hilda Anissa Nur Mauludine |
| Jenis | : | Prosedur |
| I.S | : | Jumlah pemain dan komputer belum diassign |
| F.S | : | Memanggil modul player\_number (memilih jumlah pemain) dan modul computer\_option (mengada - tiadakan komputer)  Memanggil modul level dan modul symbol |
| **Algoritma**  void multi\_player()  {  level();  player\_number();  if(player<4)  {computer\_option();}  else  {computer=false;}  symbol();  } | | |

|  |  |  |
| --- | --- | --- |
| Nama Modul | : | Help |
| Author / penanggung jawab | : | Hilda Anissa Nur Mauludine |
| Jenis | : | Prosedur |
| I.S | : | Tapilan help belum tampil |
| F.S | : | Tampilan help sudah tampil |
| **Algoritma**  void help()  {  FILE \*tutorial;  char help;  int back;  tutorial=fopen("help.TXT","r");  if (tutorial == NULL)  {  printf("File tidak dapat dibuka\n\r");  exit(1);  }  system("cls");  while((help=getc(tutorial))!=EOF)  {putch(help);}  fclose(tutorial);  SetConsoleTextAttribute(GetStdHandle(STD\_OUTPUT\_HANDLE),255);  system("pause");  exit(0);  } | | |

|  |  |  |
| --- | --- | --- |
| Nama Modul | : | Level |
| Author / penanggung jawab | : | Hilda Anissa Nur Mauludine |
| Jenis | : | Prosedur |
| I.S | : | Jumlah ular dan tangga belum diassign |
| F.S | : | Jumlah ular dan tangga telah diassign |
| **Algoritma**  void level()  {  menu=cursor(4,52,26);  switch(menu)  {  case 1: snake=3;  ladder=7;  break;  case 2: snake=5;  ladder=5;  break;  case 3: snake=7;  ladder=3;  break;  case 4: snake=7;  ladder=7;  }  } | | |

|  |  |  |
| --- | --- | --- |
| Nama Modul | : | player\_number |
| Author / penanggung jawab | : | Hilda Anissa Nur Mauludine |
| Jenis | : | Prosedur |
| I.S | : | Jumlah pemain belum diassign |
| F.S | : | Jumlah pemain sudah diassign |
| **Algoritma**  void player\_number()  {  menu=cursor(3,52,26);  switch(menu)  { case 1: player=2; break;  case 2: player=3; break;  case 3: player=4; break; }  } | | |

|  |  |  |
| --- | --- | --- |
| Nama Modul | : | computer\_option |
| Author / penanggung jawab | : | Hilda Anissa Nur Mauludine |
| Jenis | : | Prosedur |
| I.S | : | Computer belum jelas diaktifkan / nonaktifkan |
| F.S | : | Computer sudah jelas keadaannya |
| **Algoritma**  void computer\_option()  {  menu=cursor(2,55,28);  if(menu==1)  {computer=true;}  else  {computer=false;}  } | | |

|  |  |  |
| --- | --- | --- |
| Nama Modul | : | Symbol |
| Author / penanggung jawab | : | Waffi Fatur Rahman |
| Jenis | : | Prosedur |
| I.S | : | Tiap pemain belum mimilih symbol bidaknya |
| F.S | : | Tiap pemain sudah mimilih symbol bidaknya |
| **Algoritma**  void symbol()  {  int symbol[player];  for(i=0;i<player;i++)  {  keterangan=false;  while(keterangan==false)  {  menu=cursor(4,52,26);  keterangan=true;  for(j=0;j<i;j++)  {  if(menu==symbol[j])  {keterangan=false;}  }  }  switch(menu)  {  case 1: symbol[i]=1;  player\_com[i].symbol=3;  break;  case 2: symbol[i]=2;  player\_com[i].symbol=4;  break;  case 3: symbol[i]=3;  player\_com[i].symbol=5;  break;  case 4: symbol[i]=4;  player\_com[i].symbol=6;  break;  }  }  } | | |

|  |  |  |
| --- | --- | --- |
| Nama Modul | : | play\_game |
| Author / penanggung jawab | : | Waffi Fatur Rahman |
| Jenis | : | Prosedur |
| I.S | : | Posisi semua pemain berada di garis start (element ke-1 dari field) |
| F.S | : | Salah-satu pemain berada pada garis finish (element ke-100 dari field) |
| **Algoritma**  Void play\_game()  {  keterangan=false;  while(keterangan==false)  {  for(l=0;l<all\_player;l++)  {  if(player\_com[l].turn==turn)  {dice=roll\_dice(l);}  }  if(dice!=6)  {  for(l=0;l<all\_player;l++)  {  if(player\_com[l].turn==turn)  {player\_com[l].turn\_count++;}  }  turn++;  }  if(turn==5)  {turn=1;}  for(i=0;i<all\_player;i++)  {  if(player\_com[i].position==99)  {  keterangan=true;  winner=i;  }  }  }  printf("CONGRATULATION !");  if(winner<player)  {  printf("PLAYER %c WIN !",player\_com[winner].symbol);  }  else  {  printf("COMPUTER %c WIN !",player\_com[winner].symbol);  }  } | | |

|  |  |  |
| --- | --- | --- |
| Nama Modul | : | Field\_show |
| Author / penanggung jawab | : | Waffi Fatur Rahman |
| Jenis | : | Prosedur |
| I.S | : | Papan permainan tidak ditampilkan |
| F.S | : | Papan permainan ditampilkan |
| **Algoritma**  void field\_show()  {  int color\_code;  for(i=0;i<10;i++)  {  for(j=0;j<10;j++)  {  color\_code=240;  if(i%2==0)  {k=j;}  else  {k=9-j;}  Sleep(50);  {  for(l=0;l<snake;l++)  {  if(i\*10+j+1==snakefield[l][0]+1)  {color\_code=252;}  }  for(l=0;l<ladder;l++)  {  if(i\*10+j+1==ladderfield[l][0]+1)  {color\_code=249;}  }  } SetConsoleTextAttribute(GetStdHandle(STD\_OUTPUT\_HANDLE),color\_code);  gotoxy(30+6\*k,5+4\*(9-i)+0); printf("ÉÍÍÍ»\n");  gotoxy(30+6\*k,5+4\*(9-i)+1); printf("º º\n");  gotoxy(30+6\*k,5+4\*(9-i)+2); printf("ÈÍÍÍ¼\n");  gotoxy(30+6\*k,5+4\*(9-i)+3); printf(" %d \n",i\*10+j+1);  SetConsoleTextAttribute(GetStdHandle(STD\_OUTPUT\_HANDLE),240);  switch(field[i\*10+j])  {  case 3 : symbol\_color(3,249,32+6\*k,6+4\*(9-i));break;//9  case 4 : symbol\_color(4,250,32+6\*k,6+4\*(9-i));break;//10  case 5 : symbol\_color(5,252,32+6\*k,6+4\*(9-i));break;//12  case 6 : symbol\_color(6,253,32+6\*k,6+4\*(9-i));break;//13  }  }  }  } | | |

|  |  |  |
| --- | --- | --- |
| Nama Modul | : | random\_ular\_tangga |
| Author / penanggung jawab | : | Waffi Fatur Rahman |
| Jenis | : | Prosedur |
| I.S | : | Posisi ular dan tangga belum di set |
| F.S | : | Posisi ular dan tangga sudah di set |
| **Algoritma**  void random\_ular\_tangga()  {  printf("SNAKE :\n");  for(i=0;i<7;i++) {  snakefield[i][0]=(i+3)\*10+5+rand()%4;  do  {snakefield[i][1]=(i+1)\*10+5+rand()%4;}  while(snakefield[i][1]==snakefield[i-2][0]);  }  printf("LADDER :\n");  for(i=0;i<7;i++)  {  do  {ladderfield[i][0]=(i+1)\*10+rand()%5;}  while(ladderfield[i][0]==ladderfield[i-2][1]);  ladderfield[i][1]=(i+3)\*10+rand()%5;  }  } | | |

|  |  |  |
| --- | --- | --- |
| Nama Modul | : | random\_turn |
| Author / penanggung jawab | : | Waffi Fatur Rahman |
| Jenis | : | Prosedur |
| I.S | : | Urutan bermain belum ditentukan(player\_com[].turn) |
| F.S | : | Urutan bermain telah ditentukan(player\_com[].turn) |
| **Algoritma**  void random\_turn()  {  srand(time(NULL));  if(computer==1)  {all\_player=4;}  else  {all\_player=player;}  player\_com[0].turn=rand()%all\_player+1;  for(i=1;i<all\_player;i++)  {  keterangan=false;  while(keterangan==false)  {  player\_com[i].turn=rand()%all\_player+1;  keterangan=true;  for(j=i-1;j>=0;j--)  {  if(player\_com[i].turn==player\_com[j].turn)  {keterangan=false;}  }  }  }  } | | |

|  |  |  |
| --- | --- | --- |
| Nama Modul | : | random\_symbol |
| Author / penanggung jawab | : | Waffi Fatur Rahman |
| Jenis | : | Prosedur |
| I.S | : | Symbol milik computer belum di tentukan |
| F.S | : | Symbol milik computer telah di tentukan |
| **Algoritma**  void random\_symbol()  {  for(i=player;i<all\_player;i++)  {  keterangan=false;  while(keterangan==false)  {  player\_com[i].symbol=rand()%4+3;  keterangan=true;  for(j=i-1;j>=0;j--)  {  if(player\_com[i].symbol==player\_com[j].symbol)  {keterangan=false;}  }  }  }  } | | |

|  |  |  |
| --- | --- | --- |
| Nama Modul | : | Profil |
| Author / penanggung jawab | : | Waffi Fatur Rahman |
| Jenis | : | Prosedur |
| I.S | : | Informasi tiap pemain tidak di munculkan |
| F.S | : | Informasi tiap pemain sudah di munculkan |
| **Algoritma**  void profil()  {  for(i=0;i<all\_player;i++)  {  j=0;  if(i<player) {  printf("PLAYER-%d ( )",i+1);  }  else  {  printf("COM-%d ( )",i-player+1);  }  switch(player\_com[i].turn)  {  case 1: printf("FIRST - 1ST");break;  case 2: printf("SECOND - 2ND");break;  case 3: printf("THIRD - 3RD");break;  case 4: printf("FOURTH - 4TH");break;  }  printf("POSITION : %d",player\_com[i].position+1);  printf("TURN : %d",player\_com[i].turn\_count);  }  } | | |

|  |  |  |
| --- | --- | --- |
| Nama Modul | : | roll\_dice |
| Author / penanggung jawab | : | Waffi Fatur Rahman |
| Jenis | : | Prosedur |
| I.S | : | Posisi belum berpindah |
| F.S | : | Posisi sudah berpindah (melangkah sesuai dadu, naik tangga, turun ular, dimakan) |
| **Algoritma**  int roll\_dice(int turn)  {  srand(time(NULL));  dice=rand()%6+1;  field[player\_com[turn].position]=' ';  player\_com[turn].position=player\_com[turn].position+dice;  if(player\_com[turn].position>99)  {  player\_com[turn].position=198-player\_com[turn].position;  }  field[player\_com[turn].position]=player\_com[turn].symbol;  for(i=0;i<snake;i++)  {  if(snakefield[i][0]==player\_com[turn].position)  {  field[player\_com[turn].position]=' ';  player\_com[turn].position=snakefield[i][1];  field[player\_com[turn].position]=player\_com[turn].symbol;  }  }  for(i=0;i<ladder;i++)  {  if(ladderfield[i][0]==player\_com[turn].position)  {  field[player\_com[turn].position]=' ';  player\_com[turn].position=ladderfield[i][1];  field[player\_com[turn].position]=player\_com[turn].symbol;  }  }  for(i=0;i<4;i++)  {  if(player\_com[turn].position==player\_com[i].position && turn!=i)  {  player\_com[i].position=0;  field[player\_com[i].position]=player\_com[i].symbol;  }  }  return dice;  } | | |

|  |  |  |
| --- | --- | --- |
| Nama Modul | : | Exit |
| Author / penanggung jawab | : | Hilda Anissa Nur Mauludine |
| Jenis | : | Prosedur |
| I.S | : | Layar berisi hasil output terakhir |
| F.S | : | Layar kosong dan keluar program |
| **Algoritma** | | |

1. **RANCANGAN TAMPILAN**

Pada awal program di jalankan, animasi pembukaan game ular tangga akan tampil pada layar. Lalu program memunculkan beberapa pilihan sebagai menu yang akan user pilih.

Menu yang di tampilkan antara lain :

* Single player : menu permainan satu pemain dan 3 komputer
* Multi player : menu permainan dengan pemain dapat lebih dari 1 (max 4 pemain)
* Help : menu untuk menampilkan bantuan cara bermain
* Exit : menu untuk keluar dari game

Jika single player dipilih oleh user maka default 1 pemain dan 3 komputer. Lalu user akan memilih level permainan dan symbol pemain.

Jika multi player dipilih oleh user maka user akan memilih jumlah pemain, bila jumlah pemain kurang dari empat maka user di beri pilihan untuk mengadakan komputer atau tidak.

Setelah data tersebut disimpan, program akan melakukan random untuk menentukan giliran permainan. Lalu papan permainan, profil pemain, dan posisi ular serta tangga ditampilkan.

Profil pemain terdiri dari :

* Giliran pemain
* Symbol bidak pemain
* Posisi pemain
* Keterangan pemain atau komputer
* Dadu

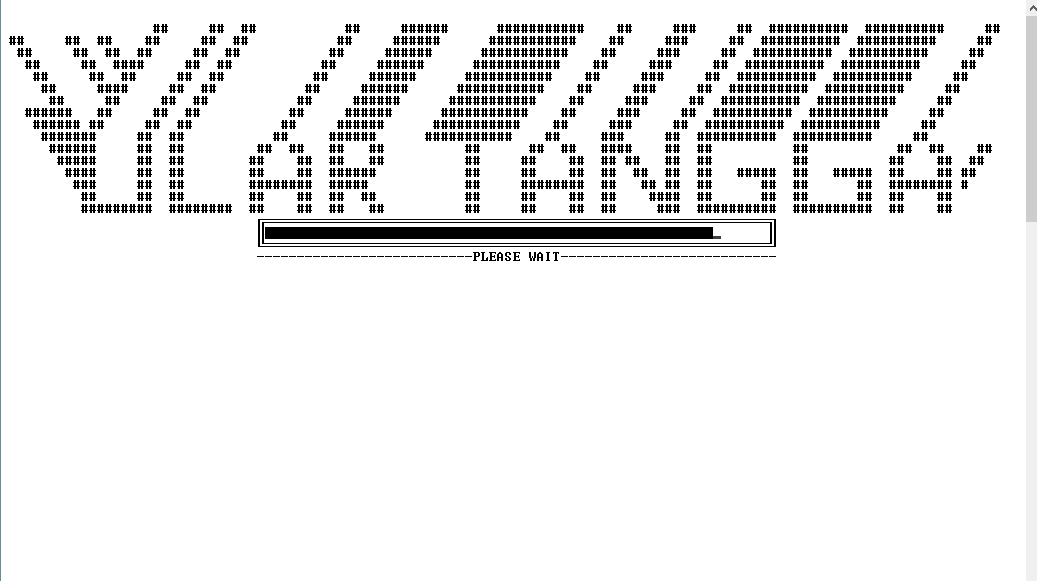
Setelah itu setiap pemain melempar dadu lalu melangkah sesuai dadu yang keluar bila dadu yang keluar adalah 6 maka user berhak melempar dadu kembali. Permainan akan terus berlangsung hingga salah satu pemain mencapai posisi finish yaitu mencapai posisi 100.

Apabila pemain menempati posisi ular atau tangga maka posisi player akan berpindah pada posisi yang sudah ditetapkan sebelumnya, posisi ular di tandai oleh warna biru sedangkan warna merah ditandai warna merah. Setiap posisi awal serta akhir ular dan tangga akan ditampikan pada keterangan di samping papan permainan.

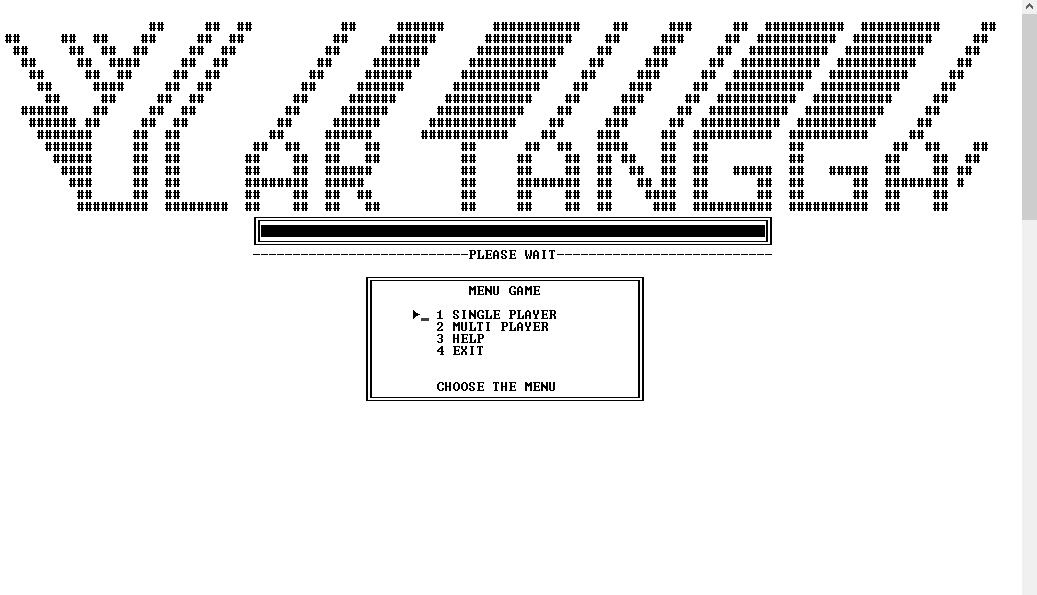
Apabila pemain menempati posisi pemain lain maka pemain yang sebelumnya berada di sana akan kembali pada posisi start. Di samping papan permainan juga di beri keterangan tentang jumlah putaran permainan yang terjadi untuk menentukan highscore.

Saat pemain mencapai finish maka permainan berakhir, jika pemain yang memenangkan game adalah user maka user di beri perintah untuk memasukan nama sebagai data pada highscore, jika pemain yang memenangkan adalah komputer maka game berakhir begitu saja.

1. **UJI COBA**
2. Tampilan awal permainan ular tangga

****

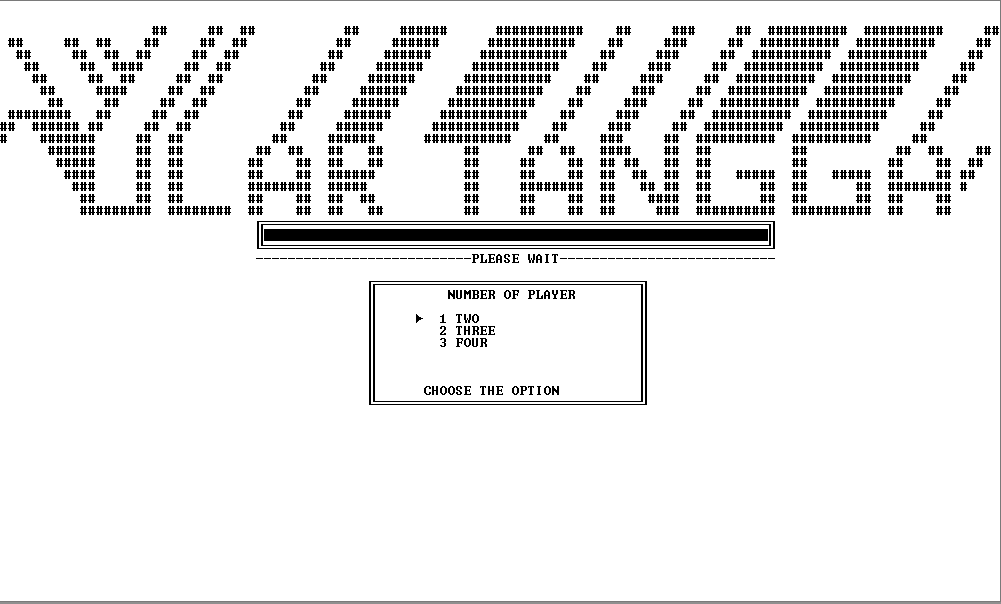
1. Tampilan Menu Game



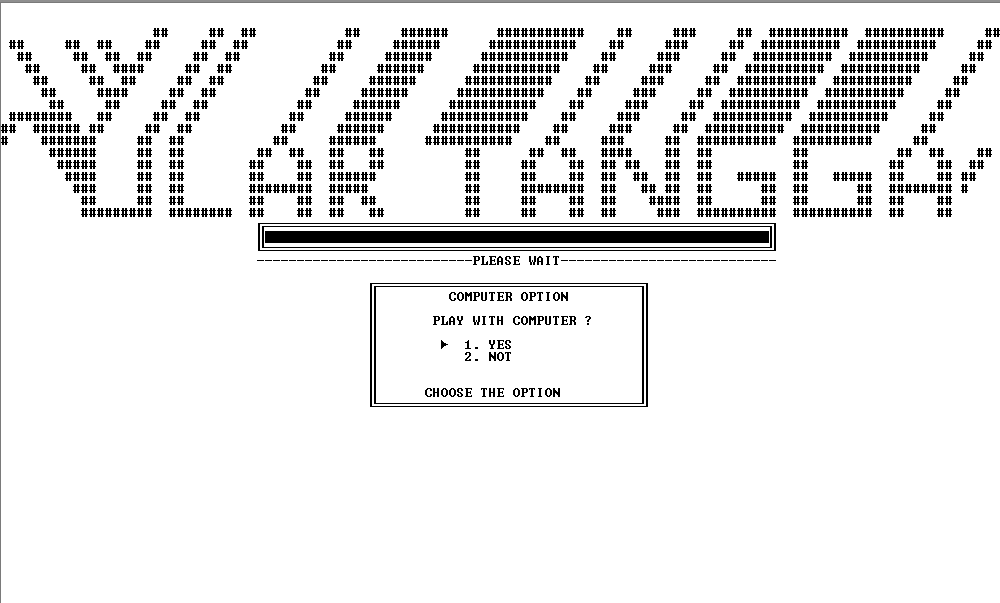
1. Tampilan Menu untuk memilih Level



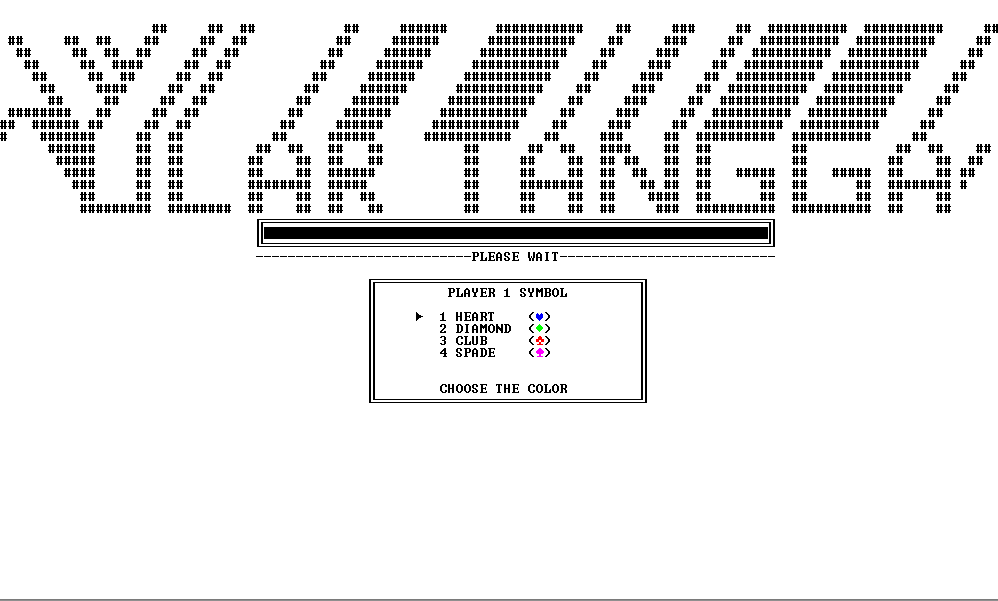
1. Tampilan untuk memilh jumlah player



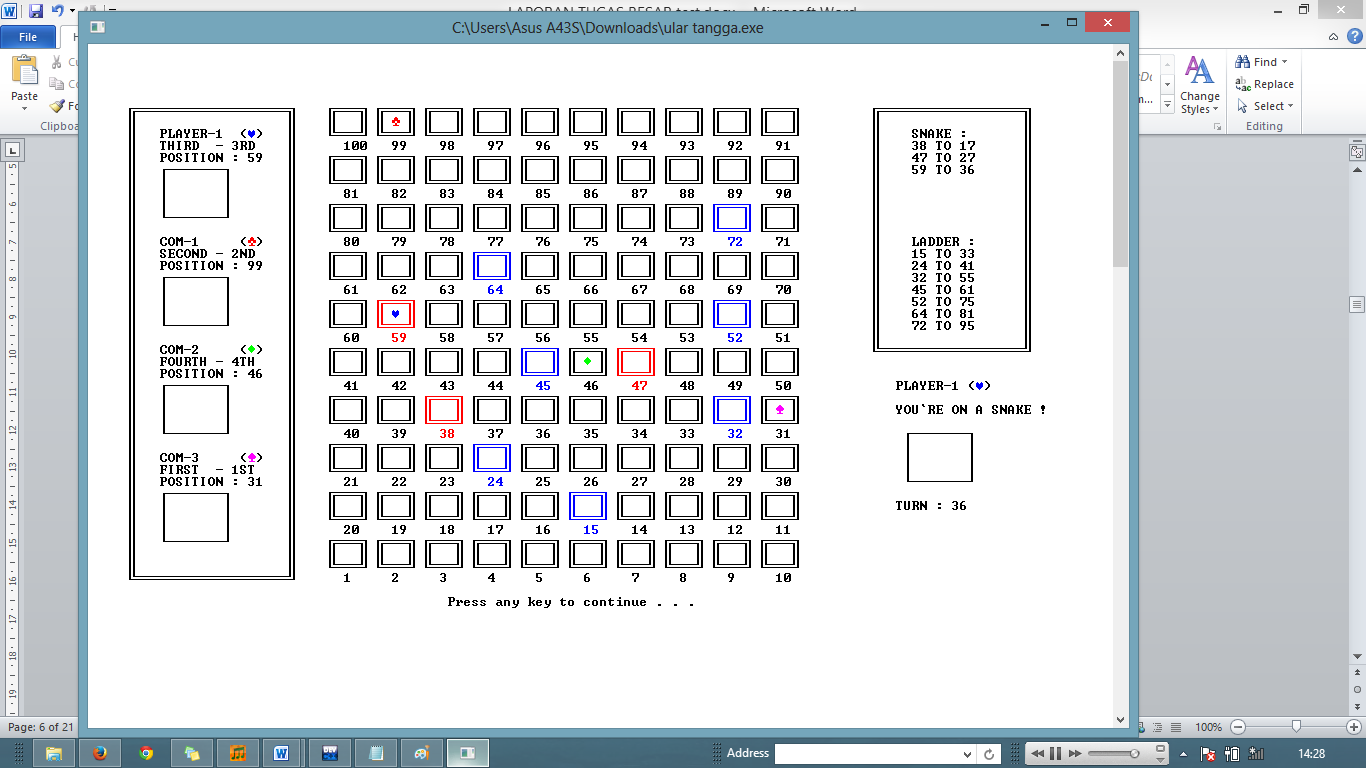
1. Tampilan untuk menentukan adanya komputer atau tidak



1. Tampilan untuk memilih symbol agar dapat membedakan para pemain



1. Tampilan papan permainan

.

1. **PEMBELAJARAN**

* Dapat mengetahui lebih banyak pemakaian library yang dibutuhkan
* Melatih softskill untuk dapat bekerja sama
* Dapat berkreasi pada tampilan program
* Mengeluarkan ide-ide dalam pembuatan design program dan algoritma program

1. **PUSTAKA**

<http://4empat.blogspot.com/2012/05/sejarah-awal-mula-permainan-ular-tangga.html>