|  |  |
| --- | --- |
| Nama Modul | main |
| Deskripsi | Program utama |
| *Author* |  |
| Jenis | Prosedur |
| *Initial State* | Program belum berjalan |
| *Final State* | Terlihat pilihan menu di layar, di mana *user* bisa memilih modul mana yang ingin dieksekusi |
| Kamus Data | |
| Pilihan: **integer** | |
| Algoritma | |
| **Do**  **write(layar)** pilihan menu yang tersedia  **input(pilihan)** pilihan menu dari *user*  **if** pilihan=1  **call** playGame()  **else if** pilihan=2  **call** showHighscore()  **else if** pilihan=3  **call** tentang()  **else if** pilihan=4  **call** bantuan()  **else if** pilihan=5  **write(layar)** ucapan selamat tinggal  **else**  **write(layar)** ucapan menu tidak tersedia  **endif**  **while** pilihan != 5 {agar setelah dari modul bisa ke menu utama lagi} | |

|  |  |
| --- | --- |
| Nama Modul | playGame |
| Deskripsi | Menyiapkan data pemain, menjalankan permainan secara keseluruhan, dan mengirimkan data *user* ke *highscore* |
| *Author* |  |
| Jenis | Prosedur |
| *Initial State* | Tampilan *main menu* pada layar |
| *Final State* | *User* dapat bermain Ludo hingga tampil kondisi game berakhir |
| Kamus Data | |
| nama: **char[25]**  lubang: **integer**  biji: **integer**  P: **pemain[2]**  Turn: **integer** | |
| Algoritma | |
| **clearscreen**  **input(name)** nama *user*  **call** generatePlayer(P,name)  Turn 🡨 suitTangan()  **If** turn=0  **write(layar)** pemberitahuan komputer mendapat giliran pertama  **else**  **write(layar)** pemberitahuan*user*mendapat giliran pertama  **endif**  **do**  **if** turn=1  **if NOT**daerahKosong(turn)  P[0].numberOfTake++  board()  **input(lubang)** nomor lubang untuk digerakkan  **if** lubang<=0 OR lubang>=8 OR pickKosong(lubang)  **clearscreen**  **else**  **call** maju(lubang,turn)  **call** board(-1)  **endif**  **else**  turn 🡨 0  **endif**  **else**  **if** **NOT**daerahKosong(turn)  **clearscreen**  **call** board(-1)  **call** srand(time(NULL))  lubang 🡨 computerPick()  **write(layar)** lubang yang dipilih oleh komputer  **call** maju(lubang,turn)  **else**  turn 🡨 1  **endif**  **endif**  **while call** isplaying()  P[0].status 🡨 WinOrLose()  **Call** saveScore()  **Call** showHighscore() | |

|  |  |
| --- | --- |
| Nama Modul | suitTangan |
| Deskripsi | Menentukan siapa yang mendapat giliran pertama |
| *Author* |  |
| Jenis | Fungsi |
| *Output* | **1 :** *user* mendapat giliran pertama  **0** : komputer mendapat giliran pertama |
| Kamus Data | |
| Opsi : **integer**  Komputer : **integer** | |
| Algoritma | |
| **do**  **if** opsi != 0  **write(layar)** input *user* salah  **endif**  **write(layar)** menu pilihan untuk suit  **input(opsi)** pilihan yang dipilih oleh *user*  **while** opsi < 1 || opsi > 5  **if** opsi!=4  komputer 🡨 suitKomputer(opsi)  **if** opsi **MOD** 3 > komputer **MOD** 3  **if** opsi = 2 **AND** komputer = 3  **return** 0  **endif**  **return** 1  **else**  **if** opsi = 3 **AND** komputer = 2  **return** 1  **endif**  **return** 0  **endif**  **else** **if** opsi = 5 {cheat jika ingin giliran kedua}  **return** 0  **else**  **return** 1 {cheat jika ingin giliran pertama}  **endif** | |

|  |  |
| --- | --- |
| Nama Modul | suitKomputer |
| Deskripsi | Menghasilkan pilihan komputer saat proses suit secara random |
| *Author* |  |
| Jenis | Fungsi |
| *Output* | **1-3** : angka yang mewakili gunting-kertas-batu |
| Kamus Data | |
| hasil : **integer** | |
| Algoritma | |
| **Call** srand(time(NULL))  **do**  hasil 🡨 rand() **MOD** 3 + 1  **while** hasil = pilihan  **depend on** hasil  **case** 1 : **write(layar)** Komputer Memilih Kertas  **case** 2: **write(layar)** Komputer Memilih Gunting  **case** 3: **write(layar)** Komputer Memilih Batu  **return** hasil | |

|  |  |
| --- | --- |
| Nama Modul | generatePlayer |
| Deskripsi | Mempersiapkan variabel pemain untuk permainan |
| *Author* |  |
| Jenis | Prosedur |
| Initial State | Variabel pemain masih kosong atau terisi bekas permainan sebelumnya |
| Final State | Variabel pemain dalam kondisi siap digunakan |
| *Input* | **P:** pemain[ ], variabel penampung data pemain  **Nama**: char[ ], variabel nama *user* |
| *Parameter Output* | **P:** pemain[ ], variabel penampung data pemain |
| Kamus Data | |
| **I** : integer | |
| Algoritma | |
| **For** i 🡨 0 to 1  **If** i=0  strcpy(P[i].nama,nama)  P[i].isComputer = false  P[i].bijiCollected = 0  P[i].numberOfShoot = 0  P[i].numberOfTake = 0  **else**  strcpy(P[i].nama,"Computer")  P[i].isComputer = true;  **endif**  P[i].bijiCollected = 0  **endfor** | |

|  |  |
| --- | --- |
| Nama Modul | board |
| Deskripsi | Membuat papan congklak pada layar dan mengisi setiap kuwuk |
| *Author* | Ilham Gibran |
| Jenis | Prosedur |
| *Initial State* | Tampilan menu utama atau tampilan papan congklak sebelumnya |
| *Final State* | Tampil papan congklak dan isi kuwuknya |
| *input* | **Lubang:** integer |
| Kamus Data | |
| papan: **char** **papan[5][25]**={{'-','-','-','9','-','-','A','-','-','B','-','-','C','-','-','D','-','-','E','-','-','F','-','-','-'},  {'/',' ','|','x','|','|','x','|','|','x','|','|','x','|','|','x','|','|','x','|','|','x','|',' ','\\'},  {'|','|','X','|',' ',' ',' ',' ',' ',' ',' ',' ','Z',' ',' ',' ',' ',' ',' ',' ',' ','|','X','|','|'},  {'\\',' ','|','x','|','|','x','|','|','x','|','|','x','|','|','x','|','|','x','|','|','x','|',' ','/'},  {'-','-','-','7','-','-','6','-','-','5','-','-','4','-','-','3','-','-','2','-','-','1','-','-','-'}, };  i,j,x,y,index,number,z,posisi: **integer**  giliran: **char[9]**  Turn: **integer** | |
| Algoritma | |
| HANDLE hdl 🡨 GetStdHandle(STD\_OUTPUT\_HANDLE)  number 🡨 10  **clearscreen**  **if** lubang>8 **AND** lubang<16  posisi 🡨 lubang - 9  **else**  posisi 🡨 14-lubang  **endif**  y 🡨 2  **for** i 🡨 0 to 4  y 🡨 y + 2  x 🡨 20  **for** j 🡨 0 to 24  x 🡨 x + 3  **call** gotoxy(x,y)  **if** papan[i][j] > 64 **AND** papan[i][j] < 71  **write(layar)** number  number++  **else** **if** papan[i][j] = 'x'  **if** index = posisi **AND** lubang != 8 **AND** lubang != 16  SetConsoleTextAttribute(hdl,6)  **endif**  **write(layar)** kewuk[index]  SetConsoleTextAttribute(hdl,15)  index++  **else** **if** papan[i][j] = 'X'  **if** z != turn **AND** z = lubang/-8 +1 **AND** lubang **MOD** 8 = 0  SetConsoleTextAttribute(hdl,6)  **endif**  **write(layar)** P[z].bijiCollected  SetConsoleTextAttribute(hdl,15)  z++  **else** **if** papan[i][j] = 'Z'  **write(layar)** hand  **else**  **write(layar)** papan[i][j]  **endif**  **endfor**  **write(layar)** \n  **endfor**    **if** turn = 1  strcpy(giliran,"Pemain")  **else**  strcpy(giliran,"Komputer")  **endif**  gotoxy(50,0) **write(layar)** giliran | |

|  |  |
| --- | --- |
| Nama Modul | computerPick |
| Deskripsi | Menentukan pilihan lubang untuk komputer |
| *Author* |  |
| Jenis | Fungsi |
| *Output* | **9-15:** nomor lubang yang akan dipilih komputer |
| Kamus Data | |
| Pick,index,realPosition,pickPosition: **integer**  Memilih: **boolean** | |
| Algoritma | |
| Memilih 🡨 false  index 🡨 9  pick 🡨 9  **while** index < 16 **AND** !memilih  realPosition 🡨 index - 9  **if** kewuk[realPosition] + index = 16  pick 🡨 index  memilih 🡨 true  **endif**  index 🡨 index + 1;  **endwhile**  index 🡨 9  **while** index < 16 **AND** !memilih  realPosition 🡨 index - 9  pickPosition 🡨 pick - 9  **if** kewuk[realPosition] > kewuk[pickPosition]  pick = index  **endif**  index 🡨 index + 1  **endwhile**  **return** pick | |

|  |  |
| --- | --- |
| Nama Modul | isPlaying |
| Deskripsi | Memeriksa apakah permainan masih berjalan atau tidak |
| *Author* |  |
| Jenis | Fungsi |
| *Output* | **playing:** boolean yang menyatakan masih atau tidak berjalannya permainan |
| Kamus Data | |
| I: **integer**  playing: **boolean** | |
| Algoritma | |
| playing 🡨 false  **while** i<14 **AND NOT**playing  **if** kewuk[i] != 0  playing 🡨 true  **endif**  i 🡨 i + 1  **endwhile**  **return** playing | |

|  |  |
| --- | --- |
| Nama Modul | WinOrLose |
| Deskripsi | Memeriksa siapa pemenang game |
| *Author* |  |
| Jenis | Fungsi |
| *Output* | Nilai integer yang menunjukan status permainan bagi *user* |
| Algoritma | |
| **If** P[0].bijiCollected > P[1].bijiCollected  **return** 1  **else** **if** P[0].bijiCollected < P[1].bijiCollected  **return** 0  **else**  **return** 2  **endif** | |

|  |  |
| --- | --- |
| Nama Modul | daerahKosong |
| Deskripsi | Memeriksa apakah deretan lubang pada bagian papan pemain yang mendapat giliran kosong atau tidak |
| *Author* |  |
| Jenis | Fungsi |
| *Output* | **kosong:** boolean yang menyatakan kosong atau tidak kuwuk |
| Kamus Data | |
| Awal,akhir: **integer**  kosong: **boolean** | |
| Algoritma | |
| kosong 🡨 true  **depend on** giliran  **case** 0 : awal 🡨 0  akhir 🡨 6  **case** 1 : awal 🡨 7  akhir 🡨 13  **default**: awal 🡨 0  **while** awal <= akhir **AND** kosong  **if** kewuk[awal] != 0  kosong 🡨 false  **endif**  awal 🡨 awal + 1  **endwhile**  **return** kosong | |

|  |  |
| --- | --- |
| Nama Modul | maju |
| Deskripsi | Menyebarkan biji pemain |
| *Author* |  |
| Jenis | Fungsi |
| *input* | Lubang**: integer**  Giliran: **integer** |
| *Iniital State* | Pemain memilih lubang yang bijinya akan disebar |
| *Final State* | Biji dari lubang tersebut telah disebar |
| Kamus Data | |
| realPosition,i,player: **integer**  out: **boolean** | |
| Algoritma | |
| Player 🡨 giliran  Out 🡨 true  **do**  **if** lubang>8 **AND** lubang<16  realPosition 🡨 lubang - 9  **else**  realPosition 🡨 14-lubang  **endif**  hand 🡨 kewuk[realPosition]  kewuk[realPosition] 🡨 0  lubang++  **while** hand > 0  **if** lubang **MOD** 8 = 0  **if** turn = 1 **AND** lubang = 8  P[0].bijiCollected 🡨 P[0].bijiCollected + 1  **else** **if** lubang = 16 **AND** turn = 0  P[1].bijiCollected 🡨 P[1].bijiCollected + 1  **else**  hand++  **endif**  **else** **if** lubang > 9  realPosition++  kewuk[realPosition] 🡨 kewuk[realPosition]+1  **else** **if** lubang = 9  realPosition 🡨 0  kewuk[realPosition] 🡨 kewuk[realPosition]+1  **else** **if** lubang = 1  realPosition🡨13  kewuk[realPosition] 🡨 kewuk[realPosition]+1  **else**  realPosition--  kewuk[realPosition] 🡨 kewuk[realPosition]+1  **endif**  hand--  **if** lubang = 16  lubang 🡨 1  **else**  lubang++  **endif**  board(lubang-1)  Sleep(900)  **endwhile**    **if** lubang-1=0  gotoxy(43,15)  **write(layar)** P[1].bijiCollected  gotoxy(45,16)  out🡨false  **else** **if** lubang-1=8  gotoxy(43,15)  **write(layar)** Posisi Terakhir biji  gotoxy(45,16)  out🡨false  **else**  **if** kewuk[realPosition] = 1  out🡨false  turn 🡨 (turn+1) **MOD** 2  **endif**  **endif**  lubang 🡨 lubang - 1  **while** out  **if** kewuk[realPosition] =1 **AND** lubang != 0 **AND** lubang != 8  tembak(lubang,player)  **if** player = 1  P[0].numberOfShoot++  **endif**  **endif** | |

|  |  |
| --- | --- |
| Nama Modul | pickKosong |
| Deskripsi | Memeriksa apa lubang yang dipilih kosong |
| *Author* |  |
| Jenis | Fungsi |
| *input* | **Pick**: integer, variabel penampung lubang yang dipilih pemain |
| *Output* | boolean yang menyatakan kosong atau tidak kuwuk |
| Kamus Data | |
| realPosition: **integer** | |
| Algoritma | |
| **if** pick > 8  realPosition🡨pick-9  **else**  realPosition🡨14-pick  **endif**  **If** kewuk[realPosition] = 0  **return** true  **else**  **return** false  **endif** | |

|  |  |
| --- | --- |
| Nama Modul | tembak |
| Deskripsi | Memeriksa apa lubang yang dipilih kosong |
| *Author* |  |
| Jenis | Prosedur |
| *Parameter input* | **Giliran:** integer, penampung nilai yang menyatakan giliran pemain |
| *Parameter Input/Output* | **Lubang:** jumlah biji pada lubang di papan congklak |
| Kamus Data | |
| realPosition: **integer**  tembak: **integer**  lumbung: **integer** | |
| Algoritma | |
| lumbung 🡨 2  **if** lubang > 8  realPosition 🡨 lubang - 9  **else**  realPosition 🡨 14-lubang  **endif**  **if** giliran = 1 **AND** lubang < 8  tembak 🡨 realPosition - 7  **if** kewuk[tembak] > 0  lumbung 🡨 0  **endif**  **else** **if** giliran = 0 **AND** lubang > 8  tembak 🡨 realPosition + 7  **if** kewuk[tembak] > 0  lumbung 🡨 1  **endif**  **else**  gotoxy(35,15)  **write(layar)** pemberitahuan tidak bisa menembak  gotoxy(45,16)  system("pause")  **endif**  **if** lumbung < 2  P[lumbung].bijiCollected 🡨 P[lumbung].bijiCollected +  kewuk[realPosition] + kewuk[tembak]  kewuk[realPosition] 🡨 0  kewuk[tembak] 🡨 0  board(-1)  gotoxy(30,15)  **write(layar)** informasi hasil menembak  gotoxy(45,18)  system("pause")  **endif** | |

|  |  |
| --- | --- |
| Nama Modul | openScore |
| Deskripsi | Membuka data highscore |
| *Author* |  |
| Jenis | Fungsi |
| *Initial State* | Variabel global penampung highscore kosong |
| *Final State* | Variabel global penampung highscore berisi data highscore |
| *output* | **Nomor:**  integer, penampung panjang data highscore |
| Algoritma | |
| **While** read(file highscore)  nomor++  **endwhile**  **return** nomor | |

|  |  |
| --- | --- |
| Nama Modul | saveScore |
| Deskripsi | Menulis data highscore |
| *Author* |  |
| Jenis | Prosedur |
| *Initial State* | Data highscore belum ditulis di file *highscore* |
| *Final State* | Data highscore ditulis di file *highscore* |
| Algoritma | |
| raw.name 🡨 P[0].nama  raw.bijiCollected 🡨 P[0].bijiCollected  raw.numberOfShoot 🡨 P[0].numberOfShoot  raw.numberOfTake 🡨 P[0].numberOfTake  raw.status 🡨 P[0].status  **write(file highscore)** raw.bijiCollected, raw.numberOfShoot,  raw.numberOfTake, raw.status | |

|  |  |
| --- | --- |
| Nama Modul | swap |
| Deskripsi | Menulis data highscore |
| *Author* |  |
| Jenis | Prosedur |
| *Initial State* | Isi variabel belum ditukar |
| *Final State* | Isi variabel ditukar |
| Algoritma | |
| Temp 🡨 show[i]  show[i] 🡨 show[j]  show[j] 🡨 temp | |

|  |  |
| --- | --- |
| Nama Modul | cekSwap |
| Deskripsi | Menghasilkan boolean untuk menentukan apa terjadinya operasi *swap* berdasarkan input *user* |
| *Author* |  |
| Jenis | Prosedur |
| *Input* | **I:** integer, nomor ruang array 1  **J**: integer**,** nomor ruang array 2  **Choice:**  integer, mewakili pilihan sort berdasarkan apa |
| *Output* | Boolean yang menentukan dilakukan atau tidak proses *swap* nilai |
| Algoritma | |
| **If** choice = 1  **return** strcmp((show[j].name),(show[i].name))<0  **else** **if** choice = 2  **return** show[j].bijiCollected>show[i].bijiCollected  **else** **if** choice = 3  **return** show[j].numberOfShoot>show[i].numberOfShoot  **else** **if** choice = 4  **return** show[j].numberOfTake<show[i].numberOfTake  **endif** | |

|  |  |
| --- | --- |
| Nama Modul | sortHighscore |
| Deskripsi | Mengatur proses sorting *highscore* |
| *Author* |  |
| Jenis | Prosedur |
| *Input* | **I:** integer, nomor ruang array 1  **J**: integer**,** nomor ruang array 2 |
| *Output* | *Highscore* ter-*sort* sesuai pilihan |
| Algoritma | |
| **For** i 🡨 0 to N-2  **for** j 🡨 i+1 to N-1  **if** cekSwap(i,j,choice)  swap(i,j)  **endif**  **endfor**  **endfor** | |

|  |  |
| --- | --- |
| Nama Modul | showHighscore |
| Deskripsi | Modul utama proses penampil *highscore* |
| *Author* |  |
| Jenis | Prosedur |
| *Initial State* | Tampilan menu utama |
| *Final State* | Tampilan *highscore* permainan |
| Kamus Data | |
| N, pilihan, i, y : **integer** | |
| Algoritma | |
| Y 🡨 14  N 🡨 openScore()  **do**  **do**  **clearscreen**  gotoxy(48,1)  **write(layar)** menu untuk sorting highscore  gotoxy(48,8)  **input(pilihan)** pilihan *user* untuk sorting highscore  **while** pilihan<1 **OR** pilihan>5  **if** pilihan!=5  sortHighscore(pilihan,N)  **write(layar)** header tabel highscore  **for** i 🡨 0 to N-1  **write(layar)** show[i].name,show[i].bijiCollected  ,show[i].numberOfShoot,show[i].numberOfTake  **if** show[i].status = 1  **write(layar) “**menang”  **else** **if** show[i].status = 2  **write(layar) “**draw“  **else**  **write(layar) “**kalah”  **endif**  y++  **endfor**  **endif**  gotoxy(48,y)  system("pause")  **while** pilihan!=5 | |

|  |  |
| --- | --- |
| Nama Modul | tentang |
| Deskripsi | Modul untuk proses *about* |
| *Author* |  |
| Jenis | Prosedur |
| *Initial State* | Tampilan menu utama |
| *Final State* | Tampilan *about* |
| Kamus Data | |
| C, tmp : **integer** | |
| Algoritma | |
| **While** (c 🡨 getc(file about)!=EOF) **AND** tmp = 0  **If** c=';'  tmp 🡨 1  **else**  putchar(c)  **endif**  **endwhile** | |

|  |  |
| --- | --- |
| Nama Modul | bantuan |
| Deskripsi | Modul untuk proses *help* |
| *Author* |  |
| Jenis | Prosedur |
| *Initial State* | Tampilan menu utama |
| *Final State* | Tampilan *help* |
| Kamus Data | |
| C, tmp : **integer** | |
| Algoritma | |
| **While** (c 🡨 getc(filr)) != EOF  **If** c=';'  tmp 🡨 1  **endif**  **if** tmp = 1  putchar(c)  **endif**  **endwhile** | |

|  |  |
| --- | --- |
| Nama Modul | gotoxy |
| Deskripsi | Memindahkan lokasi cursor ke koordinat x,y |
| *Author* |  |
| Jenis | Prosedur |
| *Initial State* | Cursor terletak pada lokasi default |
| *Final State* | Cursor terletak pada koordinat x,y |
| Kamus Data | |
| X,y : **integer,** koordinat cursor | |
| Algoritma | |
| COORD coord  coord.X 🡨 x  coord.Y 🡨 y  **call** SetConsoleCursorPosition(GetStdHandle(STD\_OUTPUT\_HANDLE), coord) | |