**LAPORAN TUGAS BESAR**

**16TIN2074 – Struktur Data dan Algoritma**

**Aplikasi Permainan Monopoli**

Logo, company name

Description automatically generated

**Disusun oleh**

|  |  |
| --- | --- |
| **Adriana Anggita Daeli** | **201524033** |
| **Pamudya Putra Pamungkas** | **201524058** |

**Program D-IV Teknik Infomatika**

**Jurusan Teknik Komputer dan Informatika**

**Politeknik Negeri Bandung**

**2020**

# DESKRIPSI APLIKASI

## Permainan Monopoli secara Umum

Monopoli merupakan permainan genre papan yang berbasis strategi. Tujuan para pemain monopoli yaitu mengumpulkan uang sewa dari lawan mereka agar lawannya bangkrut. Setiap pemain melempar dadu secara bergantian untuk memindahkan bidaknya. Hasil lemparan dadu ini akan mengarahkan bidak pemain ke satu petak, dimana pemain akan membeli petak itu atau mebayar uang sewa. Pemain monopoli diberikan sejumlah uang untuk dapat melakukan transaksi jual beli dan membayar uang sewa. Selain itu, pemain dapat pula memperdagangkan properti. Gambar 1 memperlihatkan …..

atau …. (Gambar1).



Gambar 1 Contoh papan dan bidak yang digunakan pada permainan Monopoli (sumber: [www.argos.co.uk](http://www.argos.co.uk/))

## Skenario

Permainan monopoli sendiri memiliki banyak versi tergantung pada *story board* dan aturan-aturan yang berlaku. Maka dari itu, program kami menerapkan *storyboard* beserta peraturannyasebagai berikut:

### Skenario pada Menu {Tampilan Utama Permainan}

Ketika program dieksekusi, akan tampil menu utama yang di dalamnya akan terdapat 5 opsi, yaitu :

1. **Play Game (memulai permainan Monopoli)**

*User* hanya perlu mengetik angka **1** untuk bisa mulai bermain permainan Monopoli. Ketika pemain memilih opsi **1**, pemain diminta untuk memilih karakteristik komputer dalam bermain Monopoli.

1. **Highscore**

Ketika pemain mengetik angka **2**, akan ditampilkan riwayat permainan yang masing-masing akan memiliki draft berupa:

* Nama *User*
* Menang/kalah
* Skor (Uang)
* Waktu bermain

Catatan : Semua *user* akan dimasukkan ke Highscore dan komputer tidak masuk. Urutan *user* pada highscore diurutkan dari tanggal bermain, lalu bila ada *user* yang tanggal bermainnya sama, akan diurutkan sesuai skor.

1. **Help**

Ketika pemain mengetik angka **3**, akan ditampilkan cara bermain dan aturan-aturan yang berlaku dalam permainan Monopoli.

1. **Credit**

Ketika pemain mengetik angka **4**, menampilkan data pembuat (Nama permainan, nama author, jurusan, prodi, perguruan tinggi, tahun).

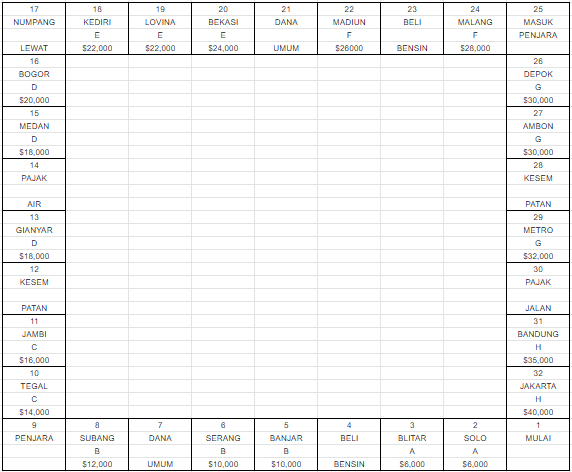
1. **Exit**

Ketika pemain mengetik angka **5**, maka program akan tertutup.

### Skenario pada Permainan Monopoli

1. **Papan**

Monopoli terdiri dari 32 petak (1 petak start, 1 petak parkir gratis, 1 petak penjara, 1 petak kantor polisi, 2 petak kesempatan, 2 petak dana umum, 2 petak pajak, 2 petak beli bensin, dan 20 petak properti).



Gambar 2

1. **Mulai Permainan**

* Pemain bermain dengan uang awal sebanyak $150.000
* Urutan bermain dimulai dari *Human* *player* terlebih dahulu.
* Posisi semua bidak pemain berada pada petak “Go” di awal permainan.

1. **Pergerakan Bidak**

Bidak pemain dapat bergerak saat pemain tersebut mendapat giliran untuk mengocok dadu. Pergerakannya ditentukan oleh hasil dadu yang dikocoknya. Setelah pemain mengocok dadu, bidak pemain akan bergerak dengan banyak langkah sebanyak hasil dari kocokkan dadu.

1. **Aksi Saat Menginjak Petak**

Setelah bidak pemain bergerak, tentunya bidak tersebut akan menginjak/berhenti pada suatu petak. Setelah bidak pemain menginjak petak, ada berbagai aksi yang bisa/harus dilakukan sesuai petak yang diinjaknya.

* **Membeli Properti**

Jika bidak pemain berhenti di properti yang belum dimiliki pemain manapun, maka pemain berhak membeli tanah sekaligus membangun rumah atau hotel seharga yang tertera di kartu hak milik, pemain dapat pula menolak untuk membeli.

* **Membayar Uang Sewa**

Jika bidak pemain berhenti di properti yang sudah dimiliki, maka pemilik properti akan menerima uang sewa seharga yang tertera pada kartu hak milik. Namun, jika properti dihipotikkan, maka tidak ada uang sewa yang harus dibayarkan.

* **Kesempatan dan Dana Umum**

Bila bidak pemain berhenti di kotak Dana Umum atau Kesempatan, maka pemain mendapatkan kartu secara acak dan mendapat tindakan sesuai yang tertulis di kartu

* **Pajak**

Jika bidak pemain berhenti di petak pajak, maka pemain harus membayar uang sejumlah yang tertera pada petak.

* **Penjara**

Pemain masuk ke dalam penjara jika

* Bidak pemain berhenti pada petak kantor polisi
* Mendapatkan angka dadu kembar sebanyak 3 kali dalam satu kali giliran. Bidak pemain tidak bergerak dahulu dan langsung masuk ke penjara ketika mendapat dadu kembar yang ketiga kalinya.

Pemain akan keluar dari penjara setelah gilirannya dilewati satu kali.

* **Parkir Gratis**

Pemain hanya berhenti sejenak hingga giliran selanjutnya. Terdapat 1 petak pada papan.

* **Gaji**

Pemain akan mendapatkan gaji sebesar $20.000 ketika menginjak atau melewati petak “Go” setelah satu kali putaran.

* **Rumah atau Hotel**

Properti dapat dibangun dengan maksimal 3 rumah dan maksimal 1 hotel. Pembangunan hotel hanya bisa dibangun setelah tanah tersebut memiliki 3 rumah. Pembangunan rumah dan hotel hanya bisa dilakukan saat pemain menginjak/berhenti pada petak tanah yang ingin dibangun tersebut.

Pembangunan rumah hanya bisa dilakukan satu kali dalam satu waktu, namun dalam satu kali giliran, pembangunan bisa dilakukan berkali-kali. Rumah dan hotel bisa dijual dengan harga setengahnya dari harga bangun.

* **Kekurangan Uang**

Ketika jumlah uang pemain kurang dari $0, maka pemain akan diberikan beberapa pilihan.

* **Menjual Properti**

Pemain hanya akan menerima separuh dari harga pokok meskipun properti tersebut telah dibangun dengan rumah atau hotel.

* **Menyatakan Bangkrut**

Maka akan dinyatakan pemain yang tidak bangkrut sebagai pemenang.

## Aturan Game

### Mulai Permainan

1. Seluruh pemain mulai dengan uang sebanyak $150.000
2. Seluruh pemain belum memiliki properti apapun
3. Seluruh pemain memulai permainan di *base start* / “Go”

### Selesai Permainan

Permainan selesai setelah tersisa satu pemain yang tidak bangkrut

1. Kondisi menang — Pemain terakhir yang bertahan ketika semua pemain lain telah menyatakan bangkrut.
2. Kondisi kalah — Pemain menyatakan dirinya bangkrut. Pemain tidak mempunyai pilihan lain selain menyatakan bangkrut ketika pemain harus membayar biaya sewa yang melebihi dari uang yang dimilikinya dan tidak ada lagi properti miliknya yang bisa dijual.

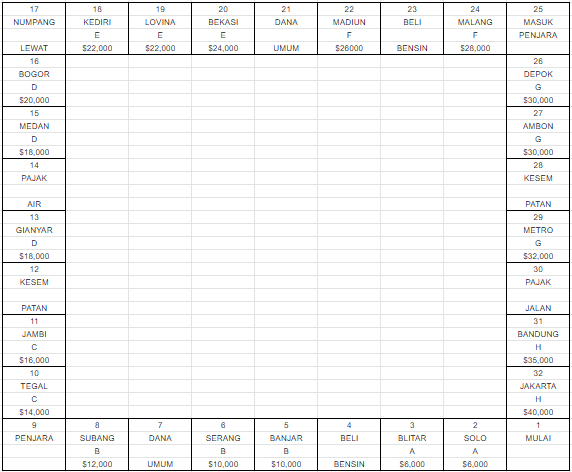
## Perlengkapan Bermain {Inisialisasi}

### Keterangan Harga

|  |  |  |
| --- | --- | --- |
| A : Solo Harga Beli $6000  Sewa Tanah $200  1 Rumah $1000  2 Rumah $3000  3 Rumah $9000  1 Rotel $16000  **Memiliki 1 komplek tanah harga sewa 2x lipat**  Harga 1 Rumah $5000  Harga 1 Hotel $5000  Harga jual $3000 | A : Blitar Harga Beli $6000  Sewa Tanah $400  1 Rumah $2000  2 Rumah $6000  3 Rumah $18000  1 Rotel $32000  **Memiliki 1 komplek tanah harga sewa 2x lipat**  Harga 1 Rumah $5000  Harga 1 Hotel $5000  Harga jual $3000 | B : Banjar Harga Beli $10000  Sewa Tanah $600  1 Rumah $3000  2 Rumah $9000  3 Rumah $27000  1 Rotel $40000  **Memiliki 1 komplek tanah harga**  **sewa 2x lipat**  Harga 1 Rumah $5000  Harga 1 Hotel $5000  Harga jual $5000 |
| B : Serang Harga Beli $10000  Sewa Tanah $600  1 Rumah $3000  2 Rumah $9000  3 Rumah $27000  1 Rotel $40000  **Memiliki 1 komplek tanah harga sewa 2x lipat**  Harga 1 Rumah $5000  Harga 1 Hotel $5000  Harga jual $5000 | B : Subang Harga Beli $12000  Sewa Tanah $800  1 Rumah $4000  2 Rumah $10000  3 Rumah $30000  1 Rotel $45000  **Memiliki 1 komplek tanah harga sewa 2x lipat**  Harga 1 Rumah $5000  Harga 1 Hotel $5000  Harga jual $6000 | C : Tegal Harga Beli $14000  Sewa Tanah $1000  1 Rumah $5000  2 Rumah $15000  3 Rumah $45000  1 Rotel $62500  **Memiliki 1 komplek tanah harga**  **sewa 2x lipat**  Harga 1 Rumah $10000  Harga 1 Hotel $10000  Harga jual $7000 |
| C : Jambi Harga Beli $16000  Sewa Tanah $1200  1 Rumah $6000  2 Rumah $18000  3 Rumah $50000  1 Rotel $70000  **Memiliki 1 komplek tanah harga sewa 2x lipat**  Harga 1 Rumah $10000  Harga 1 Hotel $10000  Harga jual $8000 | D : Gianyar Harga Beli $18000  Sewa Tanah $1400  1 Rumah $7000  2 Rumah $20000  3 Rumah $55000  1 Rotel $75000  **Memiliki 1 komplek tanah harga sewa 2x lipat**  Harga 1 Rumah $10000  Harga 1 Hotel $10000  Harga jual $9000 | D : Medan Harga Beli $18000  Sewa Tanah $1400  1 Rumah $7000  2 Rumah $20000  3 Rumah $55000  1 Rotel $75000  **Memiliki 1 komplek tanah harga**  **sewa 2x lipat**  Harga 1 Rumah $10000  Harga 1 Hotel $10000  Harga jual $9000 |

|  |  |  |
| --- | --- | --- |
| D : Bogor Harga Beli $20000  Sewa Tanah $1600  1 Rumah $8000  2 Rumah $22000  3 Rumah $60000  1 Rotel $80000  **Memiliki 1 komplek tanah harga sewa 2x lipat**  Harga 1 Rumah $10000  Harga 1 Hotel $10000  Harga jual $10000 | E : Kediri Harga Beli $22000  Sewa Tanah $1800  1 Rumah $9000  2 Rumah $25000  3 Rumah $70000  1 Rotel $87500  **Memiliki 1 komplek tanah harga sewa 2x lipat**  Harga 1 Rumah $15000  Harga 1 Hotel $15000  Harga jual $11000 | E : Lovina Harga Beli $22000  Sewa Tanah $1800  1 Rumah $9000  2 Rumah $25000  3 Rumah $70000  1 Rotel $87500  **Memiliki 1 komplek tanah harga**  **sewa 2x lipat**  Harga 1 Rumah $15000  Harga 1 Hotel $15000  Harga jual $11000 |
| E : Bekasi Harga Beli $24000  Sewa Tanah $2000  1 Rumah $10000  2 Rumah $30000  3 Rumah $75000  1 Rotel $92500  **Memiliki 1 komplek tanah harga sewa 2x lipat**  Harga 1 Rumah $15000  Harga 1 Hotel $15000  Harga jual $12000 | F : Madiun Harga Beli $26000  Sewa Tanah $2200  1 Rumah $11000  2 Rumah $33000  3 Rumah $80000  1 Rotel $97500  **Memiliki 1 komplek tanah harga sewa 2x lipat**  Harga 1 Rumah $15000  Harga 1 Hotel $15000  Harga jual $13000 | F : Malang Harga Beli $28000  Sewa Tanah $2400  1 Rumah $1200  2 Rumah $36000  3 Rumah $85000  1 Rotel $102500  **Memiliki 1 komplek tanah harga**  **sewa 2x lipat**  Harga 1 Rumah $15000  Harga 1 Hotel $15000  Harga jual $11000 |
| G : Depok Harga Beli $30000  Sewa Tanah $2600  1 Rumah $13000  2 Rumah $39000  3 Rumah $90000  1 Rotel $110000  **Memiliki 1 komplek tanah harga sewa 2x lipat**  Harga 1 Rumah $20000  Harga 1 Hotel $20000  Harga jual $15000 | G : Ambon Harga Beli $30000  Sewa Tanah $2600  1 Rumah $13000  2 Rumah $39000  3 Rumah $90000  1 Rotel $110000  **Memiliki 1 komplek tanah harga sewa 2x lipat**  Harga 1 Rumah $20000  Harga 1 Hotel $20000  Harga jual $15000 | G : Metro Harga Beli $32000  Sewa Tanah $2800  1 Rumah $15000  2 Rumah $45000  3 Rumah $100000  1 Rotel $120000  **Memiliki 1 komplek tanah harga**  **sewa 2x lipat**  Harga 1 Rumah $15000  Harga 1 Hotel $15000  Harga jual $11000 |
| H : Bandung Harga Beli $35000  Sewa Tanah $3500  1 Rumah $17000  2 Rumah $50000  3 Rumah $110000  1 Rotel $130000  **Memiliki 1 komplek tanah harga**  **sewa 2x lipat**  Harga 1 Rumah $20000  Harga 1 Hotel $20000  Harga jual $17500 | H : Jakarta Harga Beli $40000  Sewa Tanah $5000  1 Rumah $20000  2 Rumah $60000  3 Rumah $140000  1 Rotel $170000  **Memiliki 1 komplek tanah harga**  **sewa 2x lipat**  Harga 1 Rumah $20000  Harga 1 Hotel $20000  Harga jual $20000 | * Beli Bensin $1500 \* (angka dadu) * Pajak Air $7500 * Pajak Listrik $20000 |

### Keterangan Papan



### Keterangan Kartu Kesempatan dan Dana Umum

* **Kartu Kesempatan**

1. Terima uang sewa dari bank $15.000

2. Bayar perbaikan rumah $4.000 per rumah dan $15.000 per hotel

3. Maju sampai start

4. Terima bunga dari bank, terima $5.000

5. Menang Teka-Teki silang, terima $10.000

6. Bayar pajak penghasilan sebesar $15.000

7. Mundur tiga petak

8. Merusak properti umum, bayar $1.500

9. Melanggar lalu lintas, terima $5.000

10. Maju sampai Jakarta

11. Komputer anda terkena virus, bayar $5.000

* **Dana Umum**

1. Biaya dokter, bayar $1.500

2. Selamat ulang tahun, terima $1.500 dari masing-masing pemain

3. Dapat komisi, terima $5000

4. Bayar kartu $1,000 dolar atau mengambil kesempatan

5. Karena kesalahan bank, terima $10.000

6. Maju sampai Bandung

7. Sumbangan untuk bencana alam, bayar $5.000

8. Dapat sisa uang pajak jalan $5.000

9. Bayar asuransi $5.000

10. Maju sampai start

11. Dapat beasiswa, terima $10.000

## Karakteristik Komputer sebagai Pemain

* **First Timer**

1. Membeli Properti

Cara bermain dari komputer dengan karakteristik First Timer yaitu *greedy* atau ceroboh. Setiap bidak komputer menginjak tanah yang belum dimiliki *player* manapun, komputer akan langsung membelinya, selama uang yang dimilikinya masih mencukupi. Bila uang yang dimilikinya tidak mencukupi untuk membeli tanah yang sedang diinjaknya, maka komputer akan menolak untuk membelinya.

1. Membangun Rumah dan Hotel

Dalam pembangunan rumah dan hotel, setiap komputer menginjak tanah miliknya, ia akan langsung membangun 1 rumah di tanah tersebut selama uangnya mencukupi. Pembangunan rumah dilakukan setiap komputer menginjak tanah yang dimilikinya. Ketika uang yang dimilikinya tidak mencukupi untuk membangun rumah di tanah yang sedang diinjaknya, maka komputer hanya akan menyelesaikan gilirannya saja.

1. Menjual Properti

Ketika komputer menginjak tanah yang sudah dimiliki pemain lain, dan uang sewa yang harus dibayarkan melebihi dari uang yang dimilikinya saat itu, komputer akan mengutamakan penjualan tanah miliknya yang belum dibangun rumah sama sekali. Penjualan tanah dilakukan dengan memprioritaskan tanah yang paling murah dahulu sampai uangnya cukup untuk membayar uang sewa. Bila tidak ada lagi tanah miliknya yang belum dibangun rumah, maka komputer akan memprioritaskan menjual rumah di tanah yang jumlah rumahnya paling sedikit dan termurah dahulu sampai uangnya mencukupi untuk membayar hutang biaya sewa.

* **Entrepreneur**

1. Membeli Properti

Cara bermain dari komputer dengan karakteristik Entrepreneur yaitu mengedepankan modal balik. Ketika bidak komputer menginjak tanah yang belum dimiliki siapapun, komputer akan membeli tanah tersebut dan langsung membangun 1 rumah di tanah tersebut. Komputer akan melakukan pembelian dan pembangunan selama uang sisa yang akan dimilikinya setelah membeli tanah tersebut akan lebih dari $20.000, dalam hal ini, artinya komputer dengan karakeristik ini cukup waspada dan siap apabila komputer memiliki potensi menginjak tanah milik pemain lain dan sisa uang yang dimilikinya cukup untuk membayar uang sewa.

1. Membangun rumah dan hotel

Dalam pembangunan rumah dan hotel, setiap komputer menginjak tanah miliknya, ia akan langsung membangun 1 rumah di tanah tersebut selama uangnya mencukupi. Pembangunan rumah dilakukan setiap komputer menginjak tanah yang dimilikinya. Ketika uang yang dimilikinya tidak mencukupi untuk membangun rumah di tanah yang sedang diinjaknya, maka komputer hanya akan menyelesaikan gilirannya saja. Tentunya dalam hal ini, komputer akan tetap menyiapkan sisa uang lebih dari $20.000, bila membangun rumah atau hotel bisa menyebabkan sisa uang yang dimilikinya kurang dari $20.000, maka komputer tidak akan melakukan pembangunan tersebut.

1. Menjual Properti

Ketika komputer menginjak tanah yang sudah dimiliki pemain lain, dan uang sewa yang harus dibayarkan melebihi dari uang yang dimilikinya saat itu, komputer akan mengutamakan penjualan rumah di tanah yang dimilikinya dengan prioritas jumlah rumah tersedikit dan termurah. Penjualan terus dilakukan sampai uang yang dimilikinya cukup untuk membayar biaya sewa. Bila tidak ada lagi rumah yang dimilikinya, maka komputer akan memprioritaskan menjual tanah dari yang termurah dahulu.

* **Tycoon**

1. Membeli Properti

Cara bermain dari komputer dengan karakteristik Tycoon yaitu mengedepankan modal balik. Ketika bidak komputer menginjak tanah yang belum dimiliki siapapun, komputer akan membeli tanah tersebut dan langsung membangun 1 rumah di tanah tersebut. Komputer akan melakukan pembelian dan pembangunan selama uang sisa yang akan dimilikinya setelah membeli tanah tersebut akan lebih dari $30.000, dalam hal ini, artinya komputer dengan karakeristik ini cukup waspada dan siap apabila komputer memiliki potensi menginjak tanah milik pemain lain dan sisa uang yang dimilikinya cukup untuk membayar uang sewa.

1. Membangun rumah dan hotel

Dalam pembangunan rumah dan hotel, setiap komputer menginjak tanah miliknya, ia akan langsung membangun 1 rumah di tanah tersebut selama uangnya mencukupi. Pembangunan rumah dilakukan setiap komputer menginjak tanah yang dimilikinya. Ketika uang yang dimilikinya tidak mencukupi untuk membangun rumah di tanah yang sedang diinjaknya, maka komputer hanya akan menyelesaikan gilirannya saja. Tentunya dalam hal ini, komputer akan tetap menyiapkan sisa uang lebih dari $40.000, bila membangun rumah atau hotel bisa menyebabkan sisa uang yang dimilikinya kurang dari $40.000, maka komputer tidak akan melakukan pembangunan tersebut.

1. Menjual Properti

Ketika komputer menginjak tanah yang sudah dimiliki pemain lain, dan uang sewa yang harus dibayarnya melebihi dari uang yang dimilikinya saat itu, komputer akan mengutamakan penjualan rumah di tanah yang dimilikinya dengan prioritas jumlah rumah tersedikit dan termurah. Penjualan terus dilakukan sampai uang yang dimilikinya cukup untuk membayar biaya sewa. Bila tidak ada lagi rumah yang dimilikinya, maka komputer akan memprioritaskan menjual tanah dari yang termurah dahulu.

# Perancangan

## Kebutuhan Data

### Data Pemain

Data pemain digunakan untuk menampung informasi mengenai pemain. Data pemain terdiri atas *user* atau komputer, yang masing-masing memiliki data:

* PROPERTY\_COUNT = 20 (Konstanta)

**Record Enum : playerType**

* none = -1 (untuk penanda bukan pemain)
* A (untuk penanda pemain pertama)
* B (untuk penanda pemain kedua)

**Record Struct : player**

* position : integer (untuk informasi posisi pemain)
* blockOwned : integer[PROPERTY\_COUNT] (untuk informasi properti yang dimiliki pemain)
* account : integer (untuk informasi uang yang dimiliki pemain)
* skipTurn : integer (untuk informasi giliran yang akan dilewati)
* \*name : pointer of character (untuk informasi nama pemain)

### Papan Bermain

Data papan digunakan untuk menampung informasi mengenai lintasan dan detail dari masing-masing petak. Data dari masing-masing petak papan memiliki data :

**Record Enum : blocktype**

* start (untuk penanda petak start)
* property (untuk penanda petak properti)
* tax (untuk penanda petak pajak)
* punish (untuk penanda petak hukuman)
* fortune (untuk penanda petak kesempatan/dana umum)

**Record Struct : block**

* blockID : integer
* \*name : pointer of char
* price : integer
* rent : integer
* rent1 : integer
* rent2 : integer
* rent3 : integer
* rent4 : integer
* housePrice : integer
* houseOwned : integer
* type : blocktype
* owner : player
* \*next : pointer of struct block

### Skor

Data skor digunakan untuk menginput skor ke file, membaca dan menampilkan skor dari file. Data skor diantaranya adalah :

**Record Struct : score**

* date : char[26]
* time : char[10]
* name : char[10]
* enter : char
* score : integer

## Modul

### Daftar Modul

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Nama Modul | Penanggung Jawab | Jenis | Keterangan |
| 1 | help | PPP | Prosedur | Menampilkan objektif dan tata cara bermain Monopoli |
| 2 | scoreHistory | PPP | Prosedur | Menampilkan riwayat skor pemain |
| 3 | highScore | PPP | Prosedur | Menampilkan skor tertinggi yang diraih oleh pemain |
| 4 | credit | PPP | Prosedur | Menampilkan *credits* dari *author* |
| 5 | rollDice | AAD | Function | Mengocok dadu |
| 6 | buildHouse | AAD & PPP | Function | Membangun rumah |
| 7 | buyProps | AAD & PPP | Prosedur | Membeli properti |
| 8 | sellProps | AAD & PPP | Prosedur | Menjual properti |
| 9 | payRent | AAD & PPP | Prosedur | Membayar biaya sewa |
| 10 | rollFortune | AAD & PPP | Function | Mengambil kartu kesempatan/dana umum |
| 11 | afterDice | AAD & PPP | Prosedur | Menu setelah mengocok dadu |
| 12 | turn | AAD & PPP | Prosedur | Giliran |
| 13 | checkEnd | AAD & PPP | Function | Mengecek kondisi permainan |
| 14 | playGame | AAD & PPP | Prosedur | Memulai permainan Monopoli |
| 15 | entryScore | PPP | Prosedur | Memasukkan score ke file |
| 16 | gotoxy | Geeksforgeeks | Prosedur | Menunjuk pada titik tertentu |

### Spesifikasi Modul

#### help()

|  |  |
| --- | --- |
| Nama Modul | help() |
| Deskripsi | Menampilkan objektif dari permainan dan tata cara bermain Monopoli |
| Author | Pamudya Putra Pamungkas |
| Jenis | Prosedur |
| Parameter | - |
| Initial State | Tampilan masih di menu |
| Final State | Objektif dan tata cara bermain sudah tertampil |
| Kamus Data | help : char  Help.txt : FILE fhelp : file pointer |
| Algoritma | system("cls");  system("mode 130, 90");  FILE \*fhelp;  fhelp = fopen("Help.txt", "r");  int help = getc(fhelp);  while (help != EOF)  {  putchar(help);  help = getc(fhelp);  }  fclose(fhelp);  printf ("\n");  system ("pause"); |

#### scoreHistory()

|  |  |
| --- | --- |
| Nama Modul | scoreHistory() |
| Deskripsi | Menampilkan riwayat skor pemain |
| Author | Pamudya Putra Pamungkas |
| Jenis | Prosedur |
| Parameter | - |
| Initial State | Tampilan masih di menu |
| Final State | Riwayat skor sudah tertampil |
| Kamus Data | iscore, i, j : integer skor : Record score (date : char[26], time : char[10], name : char[10], enter : char,  score : int) fscore : file pointer Highscore.txt : FILE |
| Algoritma | system("cls");  FILE \*fscore;  int iscore = 1;  int i, j;  score skor;  if ((fscore = fopen("HighScore.txt", "rb")) == NULL)  {  printf("File tidak dapat dibuka\n");  exit(1);  }  gotoxy(32, 3);  for (i = 0; i < 3; i++)  {  switch (i)  {  case 0:  printf ("%c", 201); break;  case 1:  for (j = 0; j < 63; j++)  {  printf ("%c", 205);  }  break;  case 2:  printf ("%c", 187); break;  }  }  gotoxy(32, 4) ; printf("%c %c\n", 186, 186);  gotoxy(54, 4); printf ("MONOPOLY SCORE HISTORY");  gotoxy(32, 5); for (i = 0; i < 65; i++)  {  switch (i)  {  case 0:  printf ("%c", 204); break;  case 7:  printf("%c", 203); break;  case 30:  printf("%c", 203); break;  case 40:  printf("%c", 203); break;  case 64:  printf("%c", 185); break;  default :  printf("%c", 205); break;  }  }  gotoxy(32, 6); printf ("%c NO %c DATES & TIMES %c NAMES\t%c \t SCORES\t\t%c\n",186,186,186,186,186);  gotoxy(32, 7); for (i = 0; i < 65; i++)  {  switch (i)  {  case 0:  printf("%c", 204); break;  case 7:  printf("%c", 206); break;  case 30:  printf("%c", 206); break;  case 40:  printf("%c", 206); break;  case 64:  printf("%c", 185); break;  default :  printf("%c", 205); break;  }  }  printf("\n");  fscanf (fscore, "%s %s %s %d%c", &skor.date, &skor.time, &skor.name, &skor.score, &skor.enter);    gotoxy(32, 8);  i = 8;  while (!feof(fscore))  {  gotoxy(32, i);  printf ("%c %d %c %s %s %c %s\t%c\t\t\t%c\n", 186, iscore, 186, skor.date, skor.time, 186, skor.name, 186, 186);  fscanf (fscore, "%s %s %s %d%c", &skor.date, &skor.time, &skor.name, &skor.score, &skor.enter);  iscore++;  i++;  }  gotoxy(32, i);  for (i = 0; i < 65; i++)  {  switch(i){  case 0:  printf("%c", 200); break;  case 7:  printf("%c", 202); break;  case 30:  printf("%c", 202); break;  case 40:  printf("%c", 202); break;  case 64:  printf("%c", 188); break;  default :  printf("%c", 205); break;  }  }  rewind(fscore);  iscore = 0;  fscanf(fscore, "%s %s %s %d%c", &skor.date, &skor.time, &skor.name, &skor.score, &skor.enter);  while (!feof(fscore))  {  gotoxy(80, 8 + iscore);  printf("$%d", skor.score);  fscanf(fscore, "%s %s %s %d%c",&skor.date,&skor.time,&skor.name,&skor.score,&skor.enter);  iscore++;  }    gotoxy(32, 12 + iscore); system ("pause"); |

#### highScore()

|  |  |
| --- | --- |
| Nama Modul | highScore() |
| Deskripsi | Menampilkan skor tertinggi yang diraih oleh *player* |
| Author | Pamudya Putra Pamungkas |
| Jenis | Prosedur |
| Parameter | - |
| Initial State | Tampilan masih di menu |
| Final State | Highscore sudah tertampil |
| Kamus Data | i, j, tempsc : integer high : Record score (date : char[26], time : char[10], name : char[10], enter : char,  score : int)  tempdat : char[26] temptim : char[10] tempnam : char[10] fhigh : file pointer Highscore.txt : FILE |
| Algoritma | FILE \*fhigh;  int tempsc;  char tempdat[26];  char temptim[10];  char tempnam[10];  score high;  if((fhigh=fopen("HighScore.txt", "rb"))==NULL){  printf("File tidak dapat dibuka\n");  exit(1);  }  printf("\t\t\t Skor tertinggi yang dicapai sampai saat ini\n");  fscanf(fhigh, "%s %s %s %d%c",&high.date ,&high.time ,&high.name,&high.score,&high.enter);  tempsc = high.score;  strcpy(tempdat,high.date);  strcpy(temptim,high.time);  strcpy(tempnam,high.name);  while(!feof(fhigh)){  if(high.score>tempsc){  tempsc = high.score;  strcpy(tempdat,high.date);  strcpy(temptim,high.time);  strcpy(tempnam,high.name);  }  fscanf(fhigh, "%s %s %s %d%c",&high.date,&high.time, &high.name,&high.score,&high.enter);  }  printf("\t\t\t\t%d oleh %s - %s %s",tempsc,tempnam,tempdat,temptim);  fclose(fhigh); |

#### credit()

|  |  |
| --- | --- |
| Nama Modul | credit() |
| Deskripsi | Menampilkan *credits* dari *author* |
| Author | Pamudya Putra Pamungkas |
| Jenis | Prosedur |
| Parameter | - |
| Initial State | Tampilan masih di menu |
| Final State | *Credits* *author* sudah tertampil |
| Kamus Data | chr : integer fcred : file pointer Credits.txt : FILE |
| Algoritma | FILE \*fcredits;  fcredits = fopen("Credits.txt", "r");  int cred = getc(fcredits);  while(cred != EOF){  putchar(cred);  cred = getc(fcredits);  }  fclose(fcredits); |

#### rollDice()

|  |  |
| --- | --- |
| Nama Modul | rollDice() |
| Deskripsi | Mengocok dadu |
| Author | Adriana Anggita Daeli |
| Jenis | Function |
| Parameter | - |
| Initial State | Dadu belum terkocok |
| Final State | Hasil kocokan dadu sudah keluar |
| Kamus Data | i, dice : integer |
| Algoritma | int i, dice = 0;    for(i = 0; i < 2; i++) {  dice += (rand() % 6) + 1;  }  return dice; |

#### buildHouse()

|  |  |
| --- | --- |
| Nama Modul | buildHouse() |
| Deskripsi | Membangun rumah |
| Author | Adriana Anggita Daeli |
| Jenis | Function |
| Parameter | \*current\_block : block  \*activePlayer : player |
| Initial State | Belum membangun rumah |
| Final State | Rumah sudah dibangun |
| Kamus Data | flag : integer |
| Algoritma | int flag = 0;    //jika blok merukpakan milik permain  if (activePlayer->type == current\_block->owner.type) {    //jika rumah pada blok belum mencapai 4  if (current\_block->houseOwned < 4) {    //jika harga satu rumah masih lebih kurang dari uang pemain  if (current\_block->housePrice <= activePlayer->account) {    flag = 1;    activePlayer->account -= current\_block->housePrice;  current\_block->houseOwned += 1;  gotoxy(123, 28); printf("Anda punya %d rumah di %s.", current\_block->houseOwned, current\_block->name);    if(current\_block->houseOwned == 4) {  gotoxy(123, 29); printf("(Anda hanya dapat membangun maksimal 4 rumah)");  }      } else {  gotoxy(123, 28); printf("Uang and tidak mencukupi untuk membeli rumah!");    }  }  } else {  gotoxy(123, 28); printf("Anda bukan pemiliki properti ini!");  }  return flag; |

#### buyProps()

|  |  |
| --- | --- |
| Nama Modul | buyProps() |
| Deskripsi | Membeli properti |
| Author | Adriana Anggita Daeli |
| Jenis | Prosedur |
| Parameter | \*current\_block : block  \*activePlayer : player |
| Initial State | Properti belum dibeli |
| Final State | Properti sudah dibeli |
| Kamus Data | i : integer |
| Algoritma | int i;    //jika uang mecukupi  if (current\_block -> price <= activePlayer -> account)  {  activePlayer -> account -= current\_block -> price;  copy\_owner (&current\_block -> owner, activePlayer);  add\_to\_owned (activePlayer);  gotoxy(123, 23); printf("Selamat! Anda telah membeli tanah di %s.",current\_block->name);  gotoxy(123, 24); printf("Tersisa uang sebanyak $%d di akun anda.",activePlayer->account);  gotoxy (90, 4) ; printf ("$%d", activePlayer->account);    for (i = 25; i < 28; i++) {  gotoxy(123, i); printf(" ");  }  gotoxy(123, 26); system ("pause");  }  //jika uang tidak mencukupi  else {  gotoxy(123, 23); printf("Uang anda tidak cukup untuk membeli properti ini!");  for (i = 24; i < 28; i++) {  gotoxy(123, i); printf(" ");  }  gotoxy(123, 24); system ("pause");  } |

#### sellProps()

|  |  |
| --- | --- |
| Nama Modul | sellProps |
| Deskripsi | Menjual properti yang dimiliki |
| Author | Adriana Anggita Daeli |
| Jenis | Prosedur |
| Parameter | \*board : block  \*activePlayer : player |
| Initial State | Properti belum dijual |
| Final State | Properti sudah terjual |
| Kamus Data | property\_id, flag, is\_owned, i, index, counter, j : integer  \*temp : pointer of block |
| Algoritma | int property\_id, flag = 0, is\_owned =- 1, i, index, counter = 0, j;  block \*temp;  temp = (block\*)malloc(sizeof(block));  temp = board;    j = count\_properties(\*activePlayer);  if (j != 0) {  show\_player\_properties(board,\*activePlayer);  printf("0 - Back\n");  printf("Please select to sell the property:\n");  scanf("%d",&property\_id);  if (property\_id != 0) {  for(i = 0; i < PROPERTY\_COUNT; i++) {  if (activePlayer->blockOwned[i]==property\_id) {  is\_owned = 1;  index = i;  break;  }  }  if (is\_owned != -1) {  activePlayer->blockOwned[index] = -1;  player temp;  for (j = 0; j < property\_id; j++) {  board = board->next;  }  board->owner.type = none;  if (board->houseOwned == 0) {  activePlayer->account += board->price / 2;  }  if (board->houseOwned == 1) {  activePlayer->account += (board->price + board->housePrice) / 2;  }  if (board->houseOwned == 2) {  activePlayer->account += (board->price + (2 \* board->housePrice)) / 2;  }  if (board->houseOwned == 3) {  activePlayer->account += (board->price + (3 \* board->housePrice)) / 2;  }  }  board = temp;  } else {  printf("You don't have any property to sell.\n");  }  } |

#### payRent()

|  |  |
| --- | --- |
| Nama Modul | payRent() |
| Deskripsi | Membayar biaya sewa |
| Author | Adriana Anggita Daeli |
| Jenis | Prosedur |
| Parameter | \*board : block  \*activePlayer : player  \*owner : player |
| Initial State | Belum membayar biaya sewa |
| Final State | Tagihan sudah terbayar |
| Kamus Data | \*temp : pointer of block  rent, owned\_property\_count, j : integer |
| Algoritma | block \*temp;  temp = (block\*)malloc(sizeof(block));  temp = board;  int rent = 0, owned\_property\_count, j;  for(j = 0; j < activePlayer->position; j++) {  board = board->next;  }    if (board->houseOwned == 0) {  rent = board->rent;  }  if (board->houseOwned == 1) {  rent = board->rent1;  }  if (board->houseOwned == 2) {  rent = board->rent2;  }  if (board->houseOwned == 3) {  rent = board->rent3;  }  if (board->houseOwned == 4) {  rent = board->rent4;  }    if (rent > activePlayer->account) {    owned\_property\_count = count\_properties(\*activePlayer);    while(rent > activePlayer->account && owned\_property\_count != 0) {  insufficient\_fund(board,activePlayer,rent);  owned\_property\_count = count\_properties(\*activePlayer);  }  }    if (rent < activePlayer->account) {  activePlayer->account -= rent;  owner->account += rent;  } else {  activePlayer->account = 0;  board = temp;  } |

#### rollFortune()

|  |  |
| --- | --- |
| Nama Modul | rollFortune() |
| Deskripsi | Mengambil kartu kesempatan/dana umum |
| Author | Adriana Anggita Daeli |
| Jenis | Function |
| Parameter | - |
| Initial State | Belum mengambil kartu |
| Final State | Sudah mengambil aksi dari kartu yang diambil |
| Kamus Data | i : integer |
| Algoritma | int i = rand() % 6 + 1;  return i; |

#### afterDiceMenu

|  |  |
| --- | --- |
| Nama Modul | afterDiceMenu |
| Deskripsi | Memainkan permainan Monopoli baru |
| Author | Adriana Anggita Daeli & Pamudya Putra Pamungkas |
| Jenis | Prosedur |
| Parameter | \*board : block  \*activePlayer : player  \*other : player |
| Initial State | Modul belum dijalankan |
| Final State | Keluar modul |
| Kamus Data | \*temp : pointer of block  selection, j, f, id, i, insuf\_flag : integer |
| Algoritma | block \*temp;  temp = (block\*)malloc(sizeof(block));  temp = board;  int selection, j, f, id, i;  int insuf\_flag = 1;    for(j = 0; j < activePlayer -> position; j++) {  board = board -> next;  }    switch (board -> type) {    //berhenti di block properti  case property:    //print kotak keterangan  printKartu(board);    gotoxy(123, 23) ; printf("%s tiba di %s", activePlayer->name, board->name);    //jika tanah belum dimiliki  if (board -> owner.type == none) {  gotoxy(123, 24) ; printf("Apakah %s ingin membeli tanah di %s ?",activePlayer->name, board->name);  gotoxy(123, 25) ; printf("1 - Ya");  gotoxy(123, 26) ; printf("2 - Tidak");  gotoxy(123, 27) ; printf("Masukkan nomor pilihan : ");  gotoxy(148, 27) ; scanf("%d",&selection);    //jika ya membeli  if (selection == 1) {  if (activePlayer -> account < board -> price) {  while (count\_properties(\*activePlayer)!= 0 && activePlayer -> account < board -> price && insuf\_flag == 1) {  insuf\_flag = insufficient\_fund(board, activePlayer, board -> price);  }  }  buy\_property(board, activePlayer);  if (activePlayer -> type == board -> owner.type) {  house\_dialog(board, activePlayer);  }  }  }  //jika tanah orang lain  else {  pay\_rent(board, activePlayer, other);  }  break;    //berhenti di block pajak  case tax:    //print kotak keterangan  printKartu(board);    gotoxy(123, 6); printf("%s tiba di blok %s", activePlayer->name, board->type);    while (count\_properties(\*activePlayer) != 0 && activePlayer->account < board->rent) {  insuf\_flag = insufficient\_fund(board, activePlayer, board->price);    if (insuf\_flag == 0 && activePlayer->account < board->rent) {  gotoxy(123, 7); printf("Anda harus membayar tagihan!");  }  }  gotoxy(90, 4) ; printf("$%d", activePlayer->account);  if (activePlayer->account > board->rent) {  activePlayer->account = activePlayer->account - board->rent;  } else {  activePlayer->account = 0;  }  break;    //berhenti di block penjara  case punish:    //print kotak keterangan  printKartu(board);    //menunjukkan keterangan block  gotoxy(123, 6); printf("%s tiba di blok penjara", activePlayer->name, board->name);  activePlayer->skipTurn += board->rent;  break;    //berhenti di block kesempatan / dana umum  case fortune:    if (board->name == "Kesem") {  gotoxy(123, 5); printf("%s tiba di %spatan, dapat kartu. ", activePlayer->name, board->name);  } else {  gotoxy(123, 5); printf("%s tiba di %s umum, dapat kartu. ", activePlayer->name, board->name);  }  //print kotak keterangan  printKartu(board);    //acak kesempatan / dana umum bagi pemain  f = roll\_fortune();    if (f == 1) {  gotoxy(128, 1) ; printf("Dapat komisi sebanyak");  gotoxy(136, 2) ; printf ("$15000");  gotoxy(123, 7) ; printf("Uang pada akun %s bertambah", activePlayer->name);  gotoxy(123, 8) ; printf("menjadi $%d", activePlayer->account);  gotoxy(90, 4) ; printf("$%d", activePlayer->account);  activePlayer->account += 15000;  }  if (f == 2) {  gotoxy(132, 1) ; printf("Menang loteri");  gotoxy(132, 2) ; printf("terima $10000");  gotoxy(123, 7) ; printf("Uang pada akun %s bertambah", activePlayer->name);  gotoxy(123, 8) ; printf("menjadi $%d", activePlayer->account);  gotoxy(90, 4) ; printf("$%d", activePlayer->account);  activePlayer->account += 10000;  }  if (f == 3) {  gotoxy(127, 1) ; printf("Merusak properti umum");  gotoxy(129, 2) ; printf ("ganti rugi $1500");  gotoxy(123, 7) ; printf("Uang pada akun anda %s berkurang", activePlayer->name);  gotoxy(123, 8) ; printf("menjadi $%d", activePlayer->account);  gotoxy(90, 4) ; printf("$%d", activePlayer->account);  activePlayer->account -= 1500;  }  if (f == 4) {  gotoxy(130, 1) ; printf("Masuk rumah sakit");  gotoxy(133, 2) ; printf ("bayar $1500");  gotoxy(123, 7) ; printf("Uang pada akun anda %s berkurang", activePlayer->name);  gotoxy(123, 8) ; printf("menjadi $%d", activePlayer->account);  gotoxy(90, 4) ; printf("$%d", activePlayer->account);  activePlayer->account -= 1500;  }  if (f == 5) {  gotoxy(130, 1) ; printf("Memberi sumbangan");  gotoxy(131, 2) ; printf ("sebanyak $10000");  gotoxy(123, 7) ; printf("Uang pada akun anda %s berkurang", activePlayer->name);  gotoxy(123, 8) ; printf("menjadi $%d", activePlayer->account);  gotoxy(90, 4) ; printf("$%d", activePlayer->account);  activePlayer->account -= 10000;  }  if (f == 6) {  gotoxy(132, 1) ; printf("Menang loteri");  gotoxy(132, 2) ; printf("terima $5000");  gotoxy(123, 7) ; printf("Uang pada akun %s bertambah", activePlayer->name);  gotoxy(123, 8) ; printf("menjadi $%d", activePlayer->account);  gotoxy(90, 4) ; printf("$%d", activePlayer->account);  activePlayer->account += 5000;  }  break;    //kembali ke kotak start  case start:  if (board->name == "Mulai") {  gotoxy(88, 16) ; printf("Anda tiba di Stars, anda menerima $20000 dari Bank.");  gotoxy(90, 4) ; printf("$%d", activePlayer->account);  activePlayer->account += 20000;  } else if (board->name == "Hanya") {  gotoxy(88, 16) ; printf("Anda hanya lewat saja disini.");  }    } |

#### turn()

|  |  |
| --- | --- |
| Nama Modul | turn() |
| Deskripsi | Mengganti giliran |
| Author | Adriana Anggita Daeli |
| Jenis | Prosedur |
| Parameter | \*board : block  \*activePlayer : player  \*other : player |
| Initial State | Giliran belum diganti |
| Final State | Giliran sudah berganti |
| Kamus Data | \*temp : pointer of block  option, flag, next\_block, j, dice : integer |
| Algoritma | block \*temp;  int option;  temp = (block\*)malloc(sizeof(block));  temp = board;  int flag = 1, next\_block, j, dice;    while(flag) {  gotoxy (106, 25); printf ("%s", activePlayer);  gotoxy (112, 11); scanf("%d",&option);    switch(option)  {  //pilihan menu opsi kocok dadu  case 1:  if (activePlayer -> skipTurn == 0) {    //mengocok dadu  dice = rollDice();  gotoxy (88, 14); printf ("%s mengocok dadu sebesar : %d\n", activePlayer -> name, dice);    //inisialisasi block letak pemain  next\_block = (activePlayer -> position + dice);    //sampai start kembali  if (next\_block > 32) {  activePlayer -> account += 20000;  }  activePlayer -> position = next\_block % 32; //agar inisialisai letak pemain tidak lebih dari 32  after\_dice\_menu(board, activePlayer, other);    flag = 0;  board = temp;    } else {  gotoxy (88, 14); printf("Giliran anda dilewat %d kali.", activePlayer -> skipTurn);  activePlayer -> skipTurn -= 1;  flag = 0;  }  break;    //pilihan menu opsi beli rumah di block posisi pemain  case 2:  for (j = 0; j < activePlayer -> position; j++) {  board = board -> next;  }  if (board -> type == property && board -> owner.type == -1 && activePlayer -> account > board -> price) {  buy\_property (board,activePlayer);  board = temp;  } else {  printf("Tidak bisa! Anda hanya bisa membeli rumah di block yang anda miliki.\n");  }  system ("pause");  break;    case 3:  sell\_property(board,activePlayer);  break;    default:  gotoxy(127, 8); printf("Nomor salah!");    }  }  gotoxy(0, 37); system ("pause"); |

#### checkEnd()

|  |  |
| --- | --- |
| Nama Modul | checkEnd() |
| Deskripsi | Mengecek kondisi permainan selesai atau belum |
| Author | Adriana Anggita Daeli & Pamudya Putra Pamungkas |
| Jenis | Function |
| Parameter | activePlayer : player |
| Initial State | Kondisi belum dicek |
| Final State | Kondisi permainan sudah diketahui |
| Kamus Data | owned\_property\_count : integer |
| Algoritma | int owned\_property\_count = count\_properties(activePlayer);    if (activePlayer.account <= 0 && owned\_property\_count == 0) return 1;  else return 0; |

#### playGame()

|  |  |
| --- | --- |
| Nama Modul | playGame() |
| Deskripsi | Memulai permainan Monopoli |
| Author | Adriana Anggita Daeli & Pamudya Putra Pamungkas |
| Jenis | Prosedur |
| Parameter | \*board : block  \*player1 : player  \*player2 : player |
| Initial State | Permainan belum dimulai |
| Final State | Permainan telah selesai |
| Kamus Data | \*temp : pointer of block  is\_end, r, next\_block : integer |
| Algoritma | block \*temp;  temp = (block\*)malloc(sizeof(block));  temp = board;  int is\_end = 0, r, next\_block;  while (is\_end != 1) {  turn(board, player1, player2);  board = temp;  printBoard(board,\*player1,\*player2);  board = temp;  is\_end = check\_end(\*player1);  if (is\_end == 0) {  r = rollDice();    if (player2->skipTurn == 0) {  gotoxy (90, 18); printf("Computer rolled the dice: %d\n", r);  next\_block = (player2->position + r);  if (next\_block > 32) {  player2->account += 10000;  }  player2->position = next\_block % 32;  board = temp;  after\_dice\_menu\_pc(board,player2,player1);  board = temp;  } else {  printf("pc must wait %d turn\n", player2->skipTurn);  player2->skipTurn -= 1;  }    printBoard(board,\*player1,\*player2);  is\_end = check\_end(\*player2);  if (is\_end == 1){  printf("%s don't have enough money to payment, unfortunately %s lost the game!\n",player2->name,player2->name);  }  } else {  printf("%s don't have enough money to payment, unfortunately %s lost the game!\n",player1->name,player1->name);  }  } |

#### entryScore()

|  |  |
| --- | --- |
| Nama Modul | entryScore() |
| Deskripsi | Menginput skor ke file |
| Author | Pamudya Putra Pamungkas |
| Jenis | Prosedur |
| Parameter | \*player1 : player |
| Initial State | Skor belum diinput ke file |
| Final State | Skor sudah terinput ke file |
| Kamus Data | fentryscore : file pointer  date : char[20]  waktu : char[20]  space : char  t : time\_t  tm\_info : struct tm\*  Highscore.txt : FILE |
| Algoritma | FILE \*fentryscore;  char date[20];  char waktu[20];  char space='\n';  time\_t t;  struct tm\* tm\_info;  t = time(NULL);  tm\_info = localtime(&t);    //Konversi tanggal DD/MM/YYYY ke string  strftime(date,sizeof(date), "%d/%m/%Y", tm\_info);  //Konversi waktu HH:MM:SS ke string  strftime(waktu,sizeof(waktu), "%H:%M:%S", tm\_info);    fentryscore = fopen("HighScore.txt", "ab");  fprintf(fentryscore, "%s %s %s %d%c",date,waktu, player1->name,player1->account,space);  fclose(fentryscore); |

#### gotoxy()

|  |  |
| --- | --- |
| Nama Modul | entryScore() |
| Deskripsi | Menunjuk ke titik tertentu |
| Author | Pamudya Putra Pamungkas |
| Jenis | Prosedur |
| Parameter | x : integer  y : integer |
| Initial State | Belum menunjuk ke titik |
| Final State | Sudah menunjuk ke titik yang dituju |
| Kamus Data | coord : COORD |
| Algoritma | COORD coord;  coord.X = x;  coord.Y = y;  SetConsoleCursorPosition(GetStdHandle(STD\_OUTPUT\_HANDLE), coord); |

# Hasil Akhir Program

Pengerjaan laporan dan program kami dilakukan secara *offline* dan *online*. Dalam metode *online*, kami berkolaborasi menggunakan aplikasi GitHub, Google Drive, dan Line. Alamat situs GitHub kami adalah sebagai berikut.

<https://github.com/pamudyaputra/MOMON>

## Hasil Implementasi Program

### Tampilan Utama

Graphical user interface

Description automatically generated

### Tampilan *Highscore* (Skor Tertinggi)

A screenshot of a computer

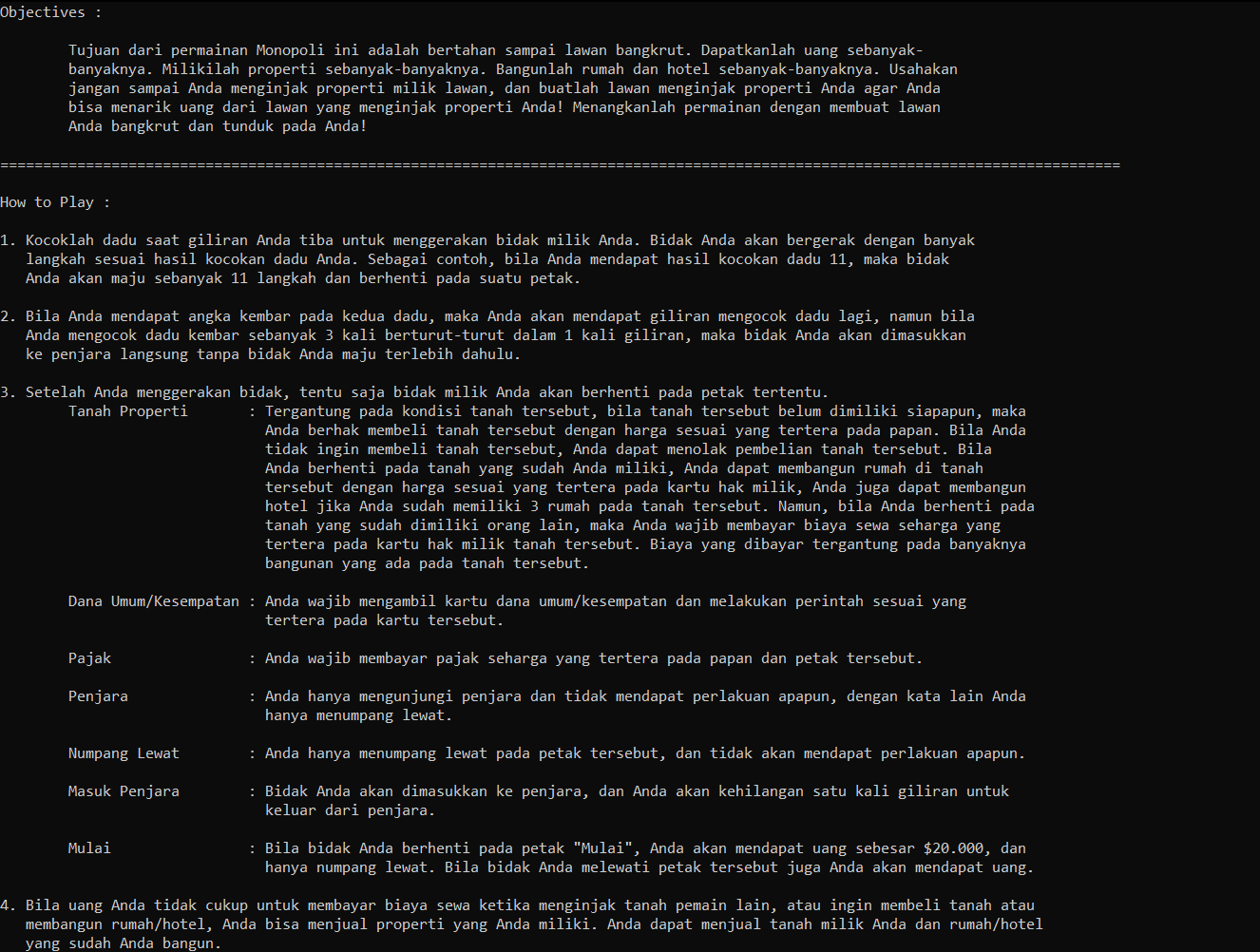
Description automatically generated with medium confidence

### Tampilan Riwayat Skor

A screenshot of a computer

Description automatically generated with medium confidence

### Tampilan Bantuan



### Tampilan *Credits*

Graphical user interface, text

Description automatically generated

### Tampilan Kocok Dadu

Graphical user interface, application

Description automatically generated

### Tampilan Papan Permainan

Graphical user interface

Description automatically generated

## Petunjuk Penggunaan

Ketika ingin menggunakan program atau memainkan permainan kami, ada beberapa hal yang perlu diperhatikan dalam penggunaannya :

1. Bukalah *file* menggunakan *compiler* berbahasa C.
2. Jalankan program.
3. *Input* angka 1 untuk memulai permainan.
4. *Input* angka 2 untuk melihat skor tertinggi yang diraih pemain sampai saat Anda menjalankan program.
5. *Input* angka 3 untuk melihat riwayat skor yang diraih oleh para pemain.
6. *Input* ankga 4 untuk melihat bantuan yang berisi objektif dari permainan dan tata cara bermain.
7. *Input* angka 5 untuk melihat *credits* dari *author*.
8. *Input* angka 6 untuk keluar dari program.

# Penutup

## *Timeline*

|  |  |
| --- | --- |
| Waktu | Kegiatan |
| 30 Desember 2020 | Pemilihan topik aplikasi |
| 8 Januari 2021 | Membeli papan Monopoli  Mulai pendeskripsian aplikasi |
| 9 Januari 2021 | Modifikasi permainan Monopoli versi Adriana & Pamudya |
| 12 Januari 2021 | Memulai perancangan  Memulai perancangan modul |
| 19 Januari 2021 | Memulai implementasi |
| 31 Januari 2021 | Pembenaran algoritma & syntax |
| 14 Februari 2021 | Penyelesaian laporan |

## Kontribusi Anggota

|  |  |
| --- | --- |
| **Adriana Anggita Daeli** | **Pamudya Putra Pamungkas** |
| Pembuatan modul Kocok Dadu | Pembuatan modul *Highscore* |
| Modifikasi modul *Highscore* | Pembuatan modul Riwayat Skor |
| Modifikasi modul Riwayat Skor | Pembuatan modul *Credits* |
| Perancangan modul utama | Pembuatan modul Bantuan |
| Perancangan modul pendukung | Perancangan modul utama |
| Implementasi | Perancangan modul pendukung |
| Penyempurnaan tampilan | Implementasi |
| Implementasi papan permainan | Pengerjaan laporan |
| Pengerjaan laporan | Penyempurnaan tampilan |

## *Lesson Learned*

### Adriana Anggita Daeli

Dalam pengerjaan tugas ini saya semakin mengerti mengenai perannya modul-modul baru yang saya temui dari sumber lain seperti gotoxy, saya pun belajar git hub yang dimana membantu kerja sama semakin mudah, dan saya pun belajar pentingnya modular programming. Bahwa dalam membuat sebuah aplikasi yang seperti game, pemanggilan suatu program tidak dapat dipastikan dan akan bergantung pada keadaan permainan yang dimana tidak dapat ditebak. Selain itu pendefinisian awal dan perancangan pun berpengaruh dalam menentukan skenario-skenario yang akan dibuat di dalam program. Saya juga paham pentingnya kerjasama dan koordinasi dengan rekan saya agar implementasi kerja kami dapat berjalan dengan baik dan tidak ada miskomunikasi.

### Pamudya Putra Pamungkas

Hal-hal yang saya dapatkan dalam pengerjaan tugas ini saya menjadi semakin paham dengan implementasi file dalam program, saya pun mempelajari modul baru yaitu gotoxy, dan saya pun belajar hal baru yaitu git hub. Dari awal hingga akhir pengerjaan tugas ini, saya belajar bahwa perencanaan awal dan aturan-aturan khusus yang akan dipakai pada program yang saya buat harus benar-benar matang agar semua skenario yang akan dijalankan dalam program dapat diimplementasikan dengan baik dan tidak terjadi kesalahpahaman. Saya pun belajar dengan adanya git hub, koordinasi dengan rekan kelompok saya semakin praktis.