

# **LAPORAN FINAL PROJECT**

## **PROGRAM ATM**

Diajukan Sebagai Tugas Mata Kuliah  
Struktur Data



Anggota Kelompok:

Ni Luh Eka Suryaningsih (2108561096)

A.A. Ngurah Frady Cakranegara (2108561097)

I Made Sudarsana Taksa Wibawa (2108561109)

**PRODI INFORMATIKA**  
**FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM**  
**UNIVERSITAS UDAYANA**  
**BUKIT JIMBARAN**  
**2022**

## **DAFTAR ISI**

<b>DAFTAR ISI .....</b>	<b>2</b>
<b>BAB I .....</b>	<b>3</b>
<b>1.1 Latar Belakang Masalah.....</b>	<b>3</b>
<b>1.2 Rumusan Masalah.....</b>	<b>3</b>
<b>1.3 Tujuan.....</b>	<b>4</b>
<b>BAB II.....</b>	<b>5</b>
<b>2.1 Rancangan Program (Flowchart) .....</b>	<b>5</b>
<b>BAB III .....</b>	<b>6</b>
<b>3.1 Coding atau Kode Program.....</b>	<b>6</b>
<b>3.2 Hasil Capture Program .....</b>	<b>22</b>
<b>KESIMPULAN .....</b>	<b>28</b>

# **BAB I**

## **PENDAHULUAN**

### **1.1 Latar Belakang Masalah**

Perkembangan teknologi dan komunikasi saat ini begitu pesat, seiring dengan pesatnya laju perkembangan ini menuntut adanya informasi yang cepat, tepat dan akurat sehingga mengakibatkan persaingan yang semakin kompetitif. Ketatnya persaingan dan pesatnya perkembangan teknologi dan informasi yang ada menuntut suatu sistem yang lebih baik, cepat dan handal dalam menyelesaikan masalah.

Di zaman yang modern pada saat sekarang ini, perkembangan teknologi perbankan sudah berkembang dengan cepat yang tujuannya memberikan pelayanan yang baik kepada nasabah dan memberikan kemudahan dalam melakukan transaksi. Ini dapat dilihat dengan semakin banyaknya dunia perbankan yang menghasilkan inovasi inovasi baru yang bermutu, contohnya seperti mesin ATM, bank digital dan lain sebagainya.

ATM merupakan sebuah mesin yang mampu untuk melakukan transaksi seperti pengiriman uang, setor tunai, cek saldo rekening, dan mengambil uang tanpa harus repot-repot mengantre di bank. Hal ini membuat ATM menjadi penunjang kebutuhan para nasabah untuk melakukan transaksi perbankan melalui media elektronik. Dalam laporan ini khusus membahas salah satu media elektronik perbankan yaitu mesin ATM dengan berbasis pemrograman berbahasa C. Pemrograman mesin ATM yang akan dilakukan melalui mesin ATM (*Automatic Teller Machine*) yaitu dapat melakukan transaksi penyimpanan, penarikan dan transfer.

### **1.2 Rumusan Masalah**

Dalam laporan ini penulis dapat merumuskan permasalahan-permasalahan yang terjadi sebagai berikut :

1. Bagaimana sistem transaksi pada mesin ATM di dalam program yang telah dibuat?

2. Bagaimana merancang sistem transaksi melalui mesin ATM pada program yang telah dibuat?
3. Bagaimana implementasi sistem transaksi pada mesin ATM pada program yang telah dibuat?

### **1.3 Tujuan**

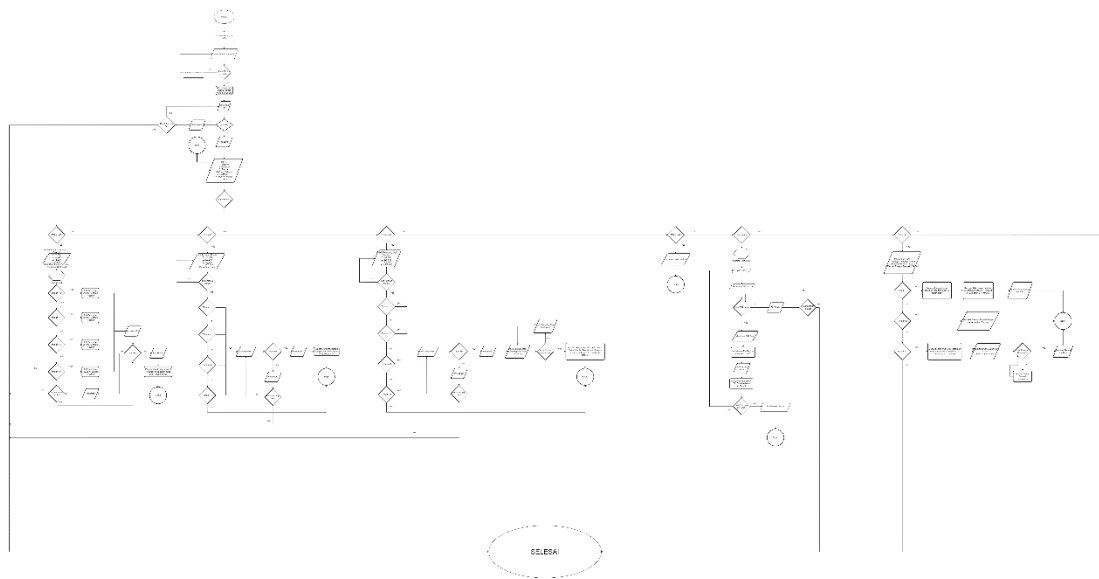
Adapun tujuan dari penulisan laporan ini adalah sebagai berikut :

1. Mengetahui sistem transaksi melalui mesin ATM di dalam program yang telah dibuat.
2. Merancang sistem transaksi melalui mesin ATM pada program yang telah dibuat.
3. Mengimplementasikan sistem transaksi melalui mesin ATM pada program yang telah dibuat.

## BAB II

### ANALISIS DAN RANCANGAN

#### 2.1 Rancangan Program (Flowchart)



[Link Flowchart](#)

## BAB III

### IMPLEMENTASI

#### 3.1 Coding atau Kode Program

Berikut adalah implementasi berupa kode program dari kasus yang diberikan

Code	Penjelasan
<pre>#include &lt;stdio.h&gt; #include &lt;string.h&gt; #include &lt;stdlib.h&gt; #include &lt;limits.h&gt; #define size 2 #define limit 50000</pre>	Library yang digunakan dalam program beserta konstanta
<pre>int hash[size]; //array untuk menyimpan nilai hash int i, j, counter = 0; //variabel bantuan int t = 0, t1 = 0; //variabel bantuan int keyTemp, keyTemp2, key; //variabel bantuan untuk hash int u = 0; //variabel bantuan int temp1, temp2; int point = 1;</pre>	Variabel – variable yang digunakan dalam program
<pre>struct stackNode{ //stack     int data;     struct stackNode *next; };  struct tempHistory{ //dipakai menyimpan nilai riwayat yang nantinya disorting pada fungsi specificHistory     int fake1;     int fake2; } tempHistory[50];  struct userData{ //menyimpan data user     int pinTemp, pin;     int temp1, temp2;     int balance;     int id;     int historyTemp, history; } userData[50];  typedef struct account{     int id;     int pin;     int balance; } account;</pre>	Deklarasi ADT (Abstract Data Type)

<pre> account acc[3];  typedef struct queueNode{     struct account data[3];     int front;     int rear; } queue;  queue antrian;  struct treeNode{     int value;     struct treeNode *left;     struct treeNode *right; } tree;  struct treeNode tree; struct treeNode* root = NULL;  struct llist{ //linked list     int data;     struct llist *next; } *head = NULL, *tail = NULL, *curr;  struct stackNode *newNode(int); int isEmpty(struct stackNode *); void push(struct stackNode **, int); int pop(struct stackNode **); int pushBack(int); </pre>	
<pre> void insertQueue(queue*, account*);  struct tree* newTreeNode(int); struct tree* insertTreeNode(struct treeNode*, int); void printTree(struct treeNode*);  void setAcc();  void initHash(); void insertHash(int);  void begin(); void search(int); void insertPin();  void deposit();  void withdraw();  void transfer(); void destinationAccount(); void searchDestination(int);  void changePin(); void newPin(); </pre>	<p><b>Deklarasi fungsi</b></p>

<pre> void checkBalance();  void history(); void recentHistory(); void allHistory(); void specificHistory();  int limitBalance(); </pre>	
<pre> void main(){     int cardNumber;     FILE *fp, *fw;     setAcc();     fp = fopen("data.txt", "r");     while(!feof(fp)){         fscanf(fp, "%d %d %d", &amp;userData[i].id, &amp;userData[i].pinTemp, &amp;userData[i].balance);         i++;     }     fclose(fp);     initHash();     for(i = 0; i &lt; size; i++){         insertHash(userData[i].id); //untuk menyimpan nilai hash     }     system("cls");     printf("===== =\\n");     printf("          ATM Program (Data Structure Final Project)      \\n");     printf("===== =\\n");     printf(" Insert your card number to continue : ");     scanf("%d", &amp;cardNumber); fflush(stdin);     search(cardNumber);     begin();     fw = fopen("dataTemp.txt", "w");     for(i = 0; i &lt; size; i++){         fprintf(fw, "%d %d %d\\n", userData[i].id, userData[i].pinTemp, userData[i].balance);     }     fclose(fw);     remove("data.txt");     rename("dataTemp.txt", "data.txt");     system("cls");     printf("===== =\\n");     printf("          Group D6 - Final Project - ATM Program      \\n");     printf("===== =\\n");     printf(" Member : \\n");     printf(" 1. Ni Luh Eka Suryaningsih          (2108561096)\\n");     printf(" 2. Anak Agung Ngurah Frady Cakra Negara (2108561097)\\n"); </pre>	<p>Fungsi main sebagai fungsi utama program ketika dijalankan</p>



<pre> printf(" 3. I Made Sudarsana Taksa Wibawa      (2108561109)\n"); printf("===== =\n\n"); } </pre>	
<pre> void begin(){     int choice;     enum choice{         DEPOSIT = 1,         WITHDRAW,         TRANSFER,         CHECKBALANCE,         CHANGEPIN,         HISTORY,         ACCOUNT,         EXIT     };     do{         system("cls");         printf("===== =====\\n");         printf("          ATM Program (Data Structure Final Project)      \\n");         printf("===== =====\\n");         printf(" 1. Deposit\\n");         printf(" 2. Withdraw\\n");         printf(" 3. Transfer\\n");         printf(" 4. Check Balance\\n");         printf(" 5. Change Pin\\n");         printf(" 6. Transfer History\\n");         printf(" 7. Account List\\n");         printf(" 8. Exit\\n");         printf("===== =====\\n");         printf(" Enter your choice (1 - 8): "); scanf("%d", &amp;choice); fflush(stdin);         system("cls");         switch(choice){             case DEPOSIT:                 deposit();                 break;             case WITHDRAW:                 withdraw();                 break;             case TRANSFER:                 transfer();                 break;             case CHECKBALANCE:                 checkBalance();                 break;             case CHANGEPIN:                 changePin();                 break;             case HISTORY: </pre>	<p>Fungsi begin digunakan untuk menampilkan menu utama program</p>

<pre> for(t = 0; t &lt; t1; t++){     pushBack(userData[t].history); } history(); break; case ACCOUNT:     system("cls");     printf("===== =====\\n");     printf("          ATM Program (Data Structure Final Project)    \\n");     printf("                      [Account] \\n");     printf("===== =====\\n");     printf(" List Account : \\n");     printTree(root);     break; case EXIT:     return;     break; default:     break; } if(choice &gt;= 1 &amp;&amp; choice &lt; 8){     printf("\\n ");     system("pause"); } }while(choice &lt; '1'    choice &gt; '8'); } </pre>	
<pre> void setAcc(){     if(fopen("data.txt", "r") == NULL){         FILE *fw;         int k;         fw = fopen("data.txt", "w");         acc[0].id = 1234;         acc[0].pin = 123456;         acc[0].balance = 500000;          acc[1].id = 4567;         acc[1].pin = 654321;         acc[1].balance = 300000;          for(k = 0; i &lt; size; i++){             fprintf(fw, "%d %d %d\\n", acc[i].id, acc[i].pin, acc[i].balance);         }         fclose(fw);     }     root = insertTreeNode(root, acc[0].id);     insertTreeNode(root, acc[1].id);      antrian.front = antrian.rear = -1;     insertQueue(&amp;antrian, &amp;acc[0]);     insertQueue(&amp;antrian, &amp;acc[1]); </pre>	<p>Fungsi setAcc digunakan untuk menginisialisasi awal data dummy dari rekening user dan menyimpan data – data tersebut pada Binary Tree dan Queue</p>

}	
<pre> void deposit(){     int choice, amount, totalDeposit;     do{         system("cls");         printf("===== =====\\n");         printf("          ATM Program (Data Structure Final Project)          \\n");         printf("                      [Deposit] \\n");         printf("===== =====\\n");         printf(" Choose the nominal amount to deposit :\\n");         printf(" 1. 20.000\\n");         printf(" 2. 50.000\\n");         printf(" 3. 75.000\\n");         printf(" 4. 100.000\\n");         printf(" 5. Back\\n");         printf("===== =====\\n");         printf(" Enter your choice (1 - 5): "); scanf("%d", &amp;choice); fflush(stdin);         if(choice &lt; 5 &amp;&amp; choice &gt; 0){             printf("\\n Enter the amount of money to deposit : "); scanf("%d", &amp;amount); fflush(stdin);         }         switch(choice){             case 1:                 totalDeposit = 20000 * amount;                 break;             case 2:                 totalDeposit = 50000 * amount;                 break;             case 3:                 totalDeposit = 75000 * amount;                 break;             case 4:                 totalDeposit = 100000 * amount;                 break;             case 5:                 return begin();                 break;             default:                 break;         }     } while(choice &lt; 1    choice &gt; 4);     system("cls");     userData[keyTemp].balance += totalDeposit;     // p = 0;     insertPin();     printf("===== =====\\n");     printf("          Transaction Successful!          \\n"); </pre>	<p>Fungsi deposit digunakan untuk memproses deposit user ke dalam rekening dengan pecahan yang telah ditentukan</p>

<pre> printf("===== \n"); printf(" Your new balance is : %d\n", userData[keyTemp].balance); userData[keyTemp].history = totalDeposit; userData[keyTemp].historyTemp = 1; t++, tl++; } </pre>	
<pre> void insertPin(){     int counter = 1;     printf("===== \n");     printf("          ATM Program (Data Structure Final Project)          \n");     printf("===== \n");     printf(" Enter your PIN to continue : ");     do{         scanf("%d", &amp;userData-&gt;pin); fflush(stdin);         if(userData-&gt;pin == userData[keyTemp].pinTemp){             printf(" PIN is correct!\n");             break;         }         else{             counter++;             printf(" PIN is incorrect!\n");             printf("\n Attempt of %d : ", counter);         }     } while(counter &lt;= 3);     if(counter &gt;= 3){         system("cls");         printf("===== =====\n");         printf("          Transaction failed!          \n");         printf("===== =====\n");         printf("\n Your account has been locked!\n\n");         exit(0);     }     system("cls"); } </pre>	<p>Fungsi insertPin digunakan untuk mengecek apakah pin yang dimasukan sudah benar atau belum</p>
<pre> void withdraw(){     int amount, choice;     do{         system("cls");         printf("===== =====\n");         printf("          ATM Program (Data Structure Final Project)          \n");         printf("          [Withdraw] \n"); </pre>	<p>Fungsi withdraw digunakan untuk memproses penarikan uang user</p>

```

printf("=====
=====\n");
printf(" Choose the nominal amount to withdraw :\n");
printf(" 1. 100.000\n");
printf(" 2. 200.000\n");
printf(" 3. 500.000\n");
printf(" 4. 1.000.000\n");
printf(" 5. Back\n");
printf("=====
=====\n");
printf(" Enter your choice (1 - 5): "); scanf("%d",
&choice); fflush(stdin);
printf("\n");
int counter = limitBalance();
if(counter){
    main();
}
switch(choice){
    case 1:
        if(userData[keyTemp].balance < 100000){
            printf(" You don't have enough money!\n");
            printf("\n ");
            system("pause");
            return withdraw();
        }
        userData[keyTemp].balance -= 100000;
        amount = 100000;
        break;
    case 2:
        if(userData[keyTemp].balance < 200000){
            printf(" You don't have enough money!\n");
            printf("\n ");
            system("pause");
            return withdraw();
        }
        userData[keyTemp].balance -= 200000;
        amount = 200000;
        break;
    case 3:
        if(userData[keyTemp].balance < 500000){
            printf(" You don't have enough money!\n");
            printf("\n ");
            system("pause");
            return withdraw();
        }
        userData[keyTemp].balance -= 500000;
        amount = 500000;
        break;
    case 4:
        if(userData[keyTemp].balance < 1000000){
            printf(" You don't have enough money!\n");
            printf("\n ");
            system("pause");
            return withdraw();
        }
        userData[keyTemp].balance -= 1000000;

```

sesuai dengan nominal yang diinginkan

<pre>         amount = 1000000;         break;     case 5:         return begin();         break;     default:         break;     } } while(choice &lt; 1    choice &gt; 5); system("cls"); insertPin(); printf("===== \n"); printf("          Transaction Successful!          \n"); printf("===== \n"); printf(" Your new balance is : %d\n", userData[keyTemp].balance); userData[t].history = amount; userData[t].historyTemp = 2; t++, t1++; } </pre>	
<pre> void transfer(){     int amount, choice, key;     do{         system("cls");         printf("===== =====\n");         printf("          ATM Program (Data Structure Final Project)          \n");         printf("          [Transfer] \n");         printf("===== =====\n");         printf(" Choose the nominal amount to transfer :\n");         printf(" 1. 100.000\n");         printf(" 2. 200.000\n");         printf(" 3. 500.000\n");         printf(" 4. 1.000.000\n");         printf(" 5. Back\n");         printf("===== =====\n");         printf(" Enter your choice (1 - 5): "); scanf("%d", &amp;choice); fflush(stdin);         printf("\n");         int counter = limitBalance();         if(counter){             main();         }         switch(choice){             case 1:                 if(userData[keyTemp].balance &lt; 100000){                     printf(" You don't have enough money!\n");                     printf("\n "); </pre>	<p>Fungsi transfer digunakan untuk memproses transfer ke rekening tujuan</p>

```

        system("pause");
        return transfer();
    }
    userData[keyTemp].balance -= 100000;
    amount = 100000;
    break;
case 2:
    if(userData[keyTemp].balance < 200000){
        printf(" You don't have enough money!\n");
        printf("\n ");
        system("pause");
        return transfer();
    }
    userData[keyTemp].balance -= 200000;
    amount = 200000;
    break;
case 3:
    if(userData[keyTemp].balance < 500000){
        printf(" You don't have enough money!\n");
        printf("\n ");
        system("pause");
        return transfer();
    }
    userData[keyTemp].balance -= 500000;
    amount = 500000;
    break;
case 4:
    if(userData[keyTemp].balance < 1000000){
        printf(" You don't have enough money!\n");
        printf("\n ");
        system("pause");
        return transfer();
    }
    userData[keyTemp].balance -= 1000000;
    amount = 1000000;
    break;
case 5:
    return begin();
    break;
default:
    break;
    }
} while(choice < 1 || choice > 5);
destinationAccount();
userData[keyTemp2].balance += amount;
system("cls");
insertPin();
printf("=====
\n");
printf("          Transaction
Successful!          \n");
printf("=====
\n");
printf(" Your new balance is : %d\n",
userData[keyTemp].balance);
userData[t].history = amount;

```

<pre>         userData[t].historyTemp = 3;         t++, t1++;     } </pre>	
<pre> void destinationAccount(){     int desId;     printf(" Enter the destination account ID : "); scanf("%d",     &amp;desId); fflush(stdin);     searchDestination(desId); } </pre>	<p>Fungsi destinationAccount digunakan untuk memasukkan rekening tujuan</p>
<pre> void searchDestination(int value){     fflush(stdin);     key = value % size;     if(hash[key] == value){         printf(" Card Number is Verified!\n");         keyTemp2 = key;     }     else{         printf(" Card Number is not Verified!\n");         return destinationAccount();     } } </pre>	<p>Fungsi digunakan untuk mencari apakah rekening tujuan cocok atau tidak</p>
<pre> void checkBalance(){     system("cls");     printf("===== =\\n");     printf("          ATM Program (Data Structure Final Project)      \\n");     printf("          [Check Balance]      \\n");     printf("===== =\\n");     printf(" Your current balance is : %d\\n", userData[keyTemp].balance); } </pre>	<p>Digunakan untuk mengecek saldo user</p>
<pre> void changePin(){     system("cls");     insertPin();     system("cls");     printf("===== =\\n");     printf("          ATM Program (Data Structure Final Project)      \\n");     printf("          [Change PIN]      \\n");     printf("===== =\\n"); </pre>	<p>Digunakan untuk mengganti pin user</p>



<pre> printf(" Enter your new PIN : "); scanf("%d", &amp;userData- &gt;temp1); fflush(stdin); printf(" Confirm your new PIN : "); scanf("%d", &amp;userData- &gt;temp2); fflush(stdin); system("cls"); if(userData-&gt;temp1 == userData-&gt;temp2){     printf("===== =====\n");     printf("                PIN Changed Successfully!                \n");     printf("===== =====\n");     userData-&gt;pinTemp = userData-&gt;temp1;     printf(" Your new PIN is : %d\n", userData-&gt;pinTemp); } else{     printf("===== =====\n");     printf("                PIN Changed Unsuccessfully!                \n");     printf("===== =====\n");     return begin(); } FILE *fw; fw = fopen("dataTemp.txt", "w"); for(i = 0; i &lt; size; i++){     fprintf(fw, "%d %d %d\n", userData[i].id, userData[i].pinTemp, userData[i].balance); } fclose(fw); remove("data.txt"); rename("dataTemp.txt", "data.txt"); } </pre>	
<pre> void history(){     int choice;     do{         system("cls");         printf("===== =====\n");         printf("                ATM Program (Data Structure Final Project)                \n");         printf("                                [History]                                 \n");         printf("===== =====\n");         printf(" 1. View Recent Transaction History\n");         printf(" 2. View All Transaction History\n");         printf(" 3. Search Transaction History\n");         printf(" 4. Back\n");         printf("===== =====\n");         printf("Enter your choice (1 - 4): "); scanf("%d", &amp;choice); fflush(stdin); </pre>	<p>Digunakan untuk menampilkan menu Riwayat transaksi</p>

<pre> switch(choice){     case 1:         recentHistory();         break;     case 2:         allHistory();         break;     case 3:         specificHistory();         break;     case 4:         return begin();         break;     default:         break; } } while(choice &lt; 1    choice &gt; 4); } </pre>	
<pre> void recentHistory(){     struct stackNode *root = NULL;     system("cls");     printf("===== =\\n");     printf("          ATM Program (Data Structure Final Project)          \\n");     printf("          [Recent History]          \\n");     printf("===== =\\n");     for(t = 0; t &lt; t1; t++){         push(&amp;root, userData[t].history);     }     for(t = t1; t &gt; t1 - 4; t--){         if(userData[t].historyTemp == 1){             printf(" Deposit : %d\\n", pop(&amp;root));         }         else if(userData[t].historyTemp == 2){             printf(" Withdraw : %d\\n", pop(&amp;root));         }         else if(userData[t].historyTemp == 3){             printf(" Transfer : %d\\n", pop(&amp;root));         }     } } </pre>	<p>Digunakan untuk melihat Riwayat transaksi terakhir yang dilakukan</p>
<pre> void allHistory(){     system("cls");     printf("===== =\\n");     printf("          ATM Program (Data Structure Final Project)          \\n");     printf("          [All History]          \\n"); </pre>	<p>Digunakan untuk melihat keseluruhan transaksi</p>

<pre> printf("===== =\n"); for(t = 0; t &lt; t1; t++){     printf(" Transaction %d\n", t + 1);     if(userData[t].historyTemp == 1){         printf(" Deposit : %d\n", userData[t].history);     }     else if(userData[t].historyTemp == 2){         printf(" Withdraw : %d\n", userData[t].history);     }     else if(userData[t].historyTemp == 3){         printf(" Transfer : %d\n", userData[t].history);     }     printf("\n"); } } </pre>	
<pre> void specificHistory(){     int category, amount;     system("cls");     printf("===== =\n");     printf("          ATM Program (Data Structure Final Project)          \n");     printf("                                [History] \n");     printf("===== =\n");     for(i = 0; i &lt; t1; i++){         tempHistory[i].fake1 = userData[i].historyTemp;         tempHistory[i].fake2 = userData[i].history;     }     for(i = 1; i &lt; t1; i++){         for(j = 0; j &lt; t1; j++){             if(tempHistory[j].fake1 &gt; tempHistory[j+1].fake1){                 temp1 = tempHistory[j].fake1;                 tempHistory[j].fake1 = tempHistory[j+1].fake1;                 tempHistory[j+1].fake1 = temp1;                  temp2 = tempHistory[j].fake2;                 tempHistory[j].fake2 = tempHistory[j+1].fake2;                 tempHistory[j+1].fake2 = temp2;             }         }     }     for(i = 0; i &lt; t1; i++){         if(tempHistory[i].fake1 == 1){             printf(" Deposit : %d\n", tempHistory[i].fake2);         }         else if(tempHistory[i].fake1 == 2){             printf(" Withdraw : %d\n", tempHistory[i].fake2);         }         else if(tempHistory[i].fake1 == 3){             printf(" Transfer : %d\n", tempHistory[i].fake2);         }     } } </pre>	<p>Digunakan untuk melihat Riwayat transaksi tertentu</p>

<pre> printf("\n"); printf(" Input history number to search\n"); printf(" Deposit : 1\n"); printf(" Withdraw : 2\n"); printf(" Transfer : 3\n"); printf(" Example : 1 100000 (Deposit with amount of 100000\n"); printf(" Enter your choice : "); scanf("%d %d", &amp;category, &amp;amount); fflush(stdin); for(i = 0; i &lt; t1; i++){     if(category == userData[i].historyTemp &amp;&amp; amount == userData[i].history){         printf(" Transaction is made on the %d transaction\n", i+1);     } } </pre>	
<pre> struct stackNode *newNode(int data) {     struct stackNode *node = (struct stackNode*)malloc(sizeof(struct stackNode));     node-&gt;data = data;     node-&gt;next = NULL;     return node; } </pre>	Digunakan untuk mendeklarasi node baru pada stack
<pre> int isEmpty(struct stackNode *root){     return !root; } </pre>	Digunakan untuk mengecek apakah stack kosong atau tidak
<pre> void push(struct stackNode **root, int data){     struct stackNode *node = newNode(data);     node-&gt;next = *root;     *root = node; } </pre>	Digunakan untuk memasukan data ke dalam stack
<pre> int pop(struct stackNode **root){     if (isEmpty(*root)){         return INT_MIN;     }     struct stackNode *temp = *root;     *root = (*root)-&gt;next;     int popped = temp-&gt;data;     free(temp);     return popped; } </pre>	Digunakan untuk menghapus data pada stack

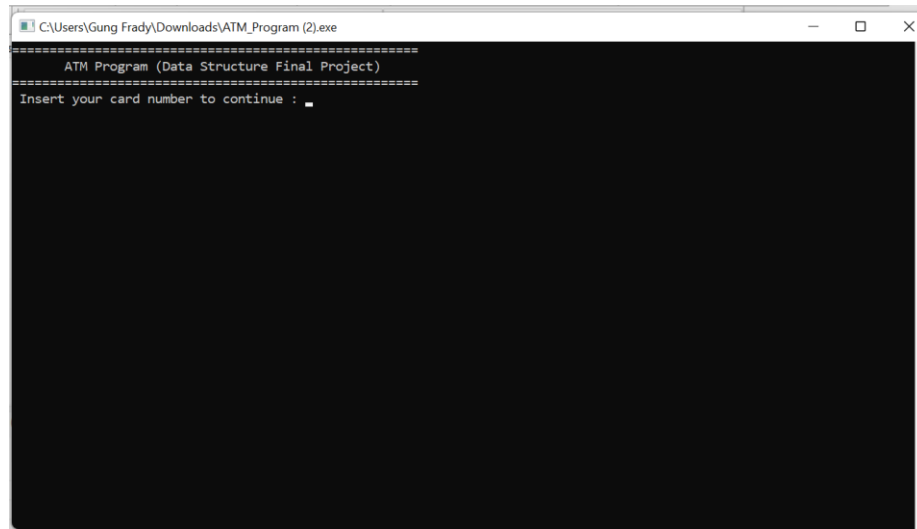
<pre> int pushBack(int value){     curr = (struct llist*)malloc(sizeof(struct llist));     curr-&gt;data = value;     if(head == NULL){         head = tail = curr;     }     else{         tail-&gt;next = curr;         tail = curr;     }     tail-&gt;next = NULL; } </pre>	<p>Digunakan untuk memasukkan data ke linked list</p>
<pre> void initHash(){     for(i = 0; i &lt; size; i++){         hash[i] = -1;     } } </pre>	<p>Digunakan untuk menginisialisasi awal hash table</p>
<pre> void insertHash(int value){     key = value % size;     printf("%d\n", key);     if(hash[key] == -1){         hash[key] = value;         printf("\n");     }     else{         printf(" Hash Collision!\n hash[%d] has element %d already!\n", key, hash[key]);         printf(" Unable to insert %d\n", value);     } } </pre>	<p>Digunakan untuk menyimpan data ke dalam hash table</p>
<pre> void search(int value){     key = value % size;     if(hash[key] == value){         printf(" Card Number is Verified!\n\n ");         keyTemp = key;         system("pause");     }     else{         printf(" Card Number is not Verified!\n\n ");         system("pause");         return main();     } } </pre>	<p>Digunakan untuk mencari kartu rekening dalam database</p>
<pre> struct tree* newTreeNode(int value){     struct treeNode *node = (struct treeNode*)malloc(sizeof(struct treeNode));     node-&gt;value = value;     node-&gt;left = NULL;     node-&gt;right = NULL;     return node; } </pre>	<p>Digunakan untuk menginisialisasi node awal dari binary tree</p>

<pre> void printTree(struct treeNode *root){     if(root == NULL){         return;     }     printTree(root-&gt;left);     printf("%d. %d \n", point, root-&gt;value);     point++;     printTree(root-&gt;right); } </pre>	Digunakan untuk mencetak isi dari binary tree
<pre> struct tree* insertTreeNode(struct treeNode *node, int value){     if (node == NULL) return newTreeNode(value);     if (value &lt; node-&gt;value) {         node-&gt;left = insertTreeNode(node-&gt;left, value);     }     else if (value &gt; node-&gt;value) {         node-&gt;right = insertTreeNode(node-&gt;right, value);     }     return node; } </pre>	Digunakan untuk memasukkan data ke dalam binary tree
<pre> void insertQueue(queue *q, account *acc){     if(q-&gt;front == NULL){         q-&gt;front = q-&gt;rear = acc;     }     else{         q-&gt;rear = acc;     } } </pre>	Digunakan untuk memasukkan data ke dalam queue
<pre> int limitBalance(){     printf("\n");     if(userData[keyTemp].balance &lt;= limit){         printf(" Your balance is below the limit!\n\n");         printf(" Your balance is %d\n\n ", userData[keyTemp].balance);         system("pause");         return 1;     } } </pre>	Digunakan untuk membatasi jumlah minimum uang yang harus ada pada rekening user

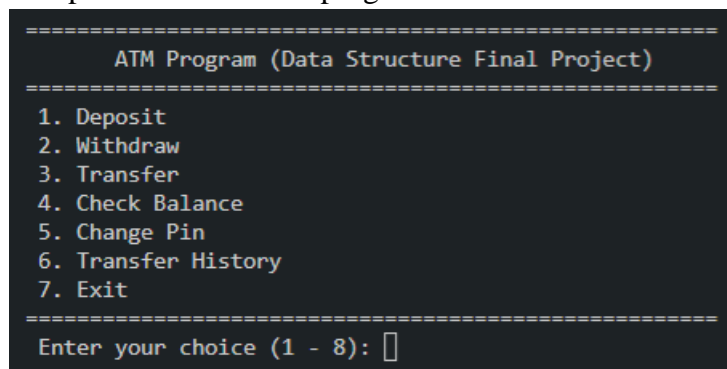
### 3.2 Hasil Capture Program

Berikut adalah beberapa capture hasil running kode program di atas.

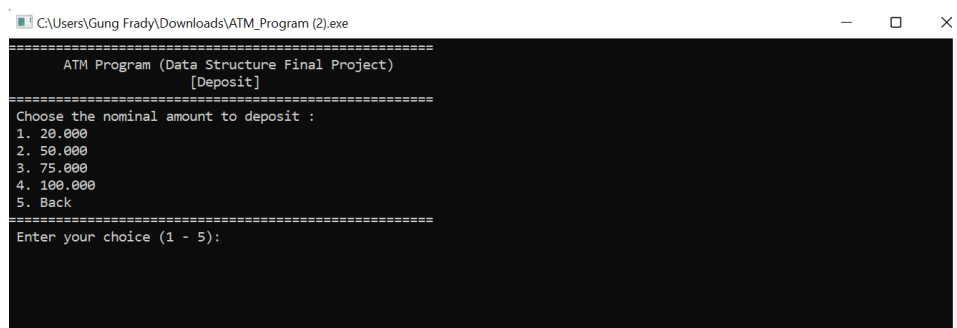
#### 1. Tampilan awal setelah dirunning



## 2. Tampilan awal di menu program



## 3. Menu Deposit




```
=====
                        ATM Program (Data Structure Final Project)
                        [Deposit]
=====
Choose the nominal amount to deposit :
1. 20.000
2. 50.000
3. 75.000
4. 100.000
5. Back
=====
Enter your choice (1 - 5): 4

Enter the amount of money to deposit : 10
```

```
=====
                        Transaction Successful!
=====
Your new balance is : 1500000

Press any key to continue . . .
```

#### 4. Menu Withdraw



```
=====
                        ATM Program (Data Structure Final Project)
                        [Withdraw]
=====
Choose the nominal amount to withdraw :
1. 100.000
2. 200.000
3. 500.000
4. 1.000.000
5. Back
=====
Enter your choice (1 - 5):
```

```
=====
                        Transaction Successful!
=====
Your new balance is : 400000

Press any key to continue . . .
```

#### 5. Menu Transfer



```
C:\Users\Gung Frady\Downloads\ATM_Program (2).exe

=====
      ATM Program (Data Structure Final Project)
      [Transfer]
=====
Choose the nominal amount to transfer :
1. 100.000
2. 200.000
3. 500.000
4. 1.000.000
5. Back
=====
Enter your choice (1 - 5): 1
```

```
=====
      ATM Program (Data Structure Final Project)
      [Transfer]
=====
Choose the nominal amount to transfer :
1. 100.000
2. 200.000
3. 500.000
4. 1.000.000
5. Back
=====
Enter your choice (1 - 5): 1

Enter the destination account ID : 4567
```

```
=====
      Transaction Successful!
=====
Your new balance is : 300000

Press any key to continue . . .
```

## 6. Menu Check Balance

```
C:\Users\Gung Frady\Downloads\ATM_Program (2).exe

=====
      ATM Program (Data Structure Final Project)
      [Check Balance]
=====
Your current balance is : 500000

Press any key to continue . . .
```

## 7. Menu Transfer History

```
C:\Users\Gung Frady\Downloads\ATM_Program (2).exe

=====
ATM Program (Data Structure Final Project)
[History]
=====
1. View Recent Transaction History
2. View All Transaction History
3. Search Transaction History
4. Back
=====
Enter your choice (1 - 4):
```

```
=====
ATM Program (Data Structure Final Project)
[Recent History]
=====
Transfer : 100000
Withdraw : 100000

Press any key to continue . . .
```

```
=====
ATM Program (Data Structure Final Project)
[All History]
=====
Transaction 1
Withdraw : 100000

Transaction 2
Transfer : 100000

Press any key to continue . . .
```

```
=====
ATM Program (Data Structure Final Project)
[History]
=====
Withdraw : 100000

Input history number to search
Deposit : 1
Withdraw : 2
Transfer : 3
Example : 1 100000 (Deposit with amount of 100000)
Enter your choice :
```

```
=====
                        ATM Program (Data Structure Final Project)
                        [History]
=====

Withdraw : 100000
Transfer : 100000

Input history number to search
Deposit : 1
Withdraw : 2
Transfer : 3
Example : 1 100000 (Deposit with amount of 100000
Enter your choice : 3 100000
Transaction is made on the 2 transaction

Press any key to continue . . .
```

## 8. Exit



```
C:\Users\Gung Frady\Downloads\ATM_Program (2).exe

Press any key to continue . . .
```

## **BAB IV**

### **KESIMPULAN**

Dalam program bahasa C terdapat banyak jenis materi yang dapat kita pelajari dan kembangkan. Khususnya pada mata kuliah Struktur Data ini yaitu seperti penggunaan struct/ADT, Stack dan Queue, Sorting, Searching, Array, Linked List, Hashing, dan masih banyak lagi. Pada penugasan ini kami mendapatkan kasus pertama yaitu “**Program mesin ATM**”. Pada program yang kami buat sudah cukup sesuai dengan kasus yang diberikan dan sudah mengimplementasikan materi-materi yang disyaratkan harus ada dalam program, seperti penggunaan, array, linked list, ADT, Stack dan Queue (untuk penambahan data dan antrian), sorting (untuk pengurutan riwayat transaksi berdasarkan beberapa kunci), searching (mencari riwayat transaksi yang diinginkan), Hashing (pada penyimpanan data rekening), rekursif pada saat mencetak data tree, serta penggunaan binary tree untuk menyimpan data dari rekening user.

Cara kerja program juga sudah sesuai, dimana terdapat 3 menu utama, deposit, withdraw, dan transfer. Deposit digunakan untuk mendepositkan uang ke dalam rekening user, withdraw untuk menarik uang sesuai saldo dari rekening user, dan transfer digunakan untuk mengirimkan uang ke rekening tujuan yang dipilih oleh user.