

# ADVANCED PROGRAMMING ASSIGNMENT

```

1
var atpos=inputs[i].indexOf("@");
var dotpos=inputs[i].lastIndexOf(".");
if (atpos<1 || dotpos<atpos+2 || dotpos>inputs[i].length-1)
document.getElementById("errEmail").innerHTML = "Invalid email address";
else
document.getElementById(div).innerHTML = "Valid";
}
use if (i==5)

```

**Damian Malinki**  
**Systems Development**  
**Gauteng City College**

**Systems Development**  
**Gauteng City College**

Gauteng City College

```
1 package com.mycompany.bankingapplication1;
2
3
4 import java.util.ArrayList;
5 import java.util.HashMap;
6 import java.util.Scanner;
7
8 class BankAccount {
9     private final String accountNumber;
10    private final String accountHolderName;
11    private double balance;
12    private final ArrayList<String> transactions;
13
14    public BankAccount(String accountNumber, String accountHolderName, double initialBalance) {
15        this.accountNumber = accountNumber;
16        this.accountHolderName = accountHolderName;
17        this.balance = initialBalance;
18        this.transactions = new ArrayList<>();
19        transactions.add("Account created with initial balance: " + initialBalance);
20    }
21
22    public String getAccountNumber() {
23        return accountNumber;
24    }
25
26    public String getAccountHolderName() {
27        return accountHolderName;
28    }
29
30    public double getBalance() {
31        return balance;
32    }
33
34    public ArrayList<String> getTransactions() {
35        return transactions;
36    }
37
38    public void deposit(double amount) {
39        if (amount > 0) {
40            balance += amount;
41            transactions.add("Deposited: " + amount);
42        } else {
43            System.out.println("Invalid deposit amount.");
44        }
45    }
46 }
```

com.mycompany.bankingapplication1.BankAccount

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help BankingApplication1 - Apache NetBeans IDE 21 Search (Ctrl+I)

<default config> 267.8/376.0MB

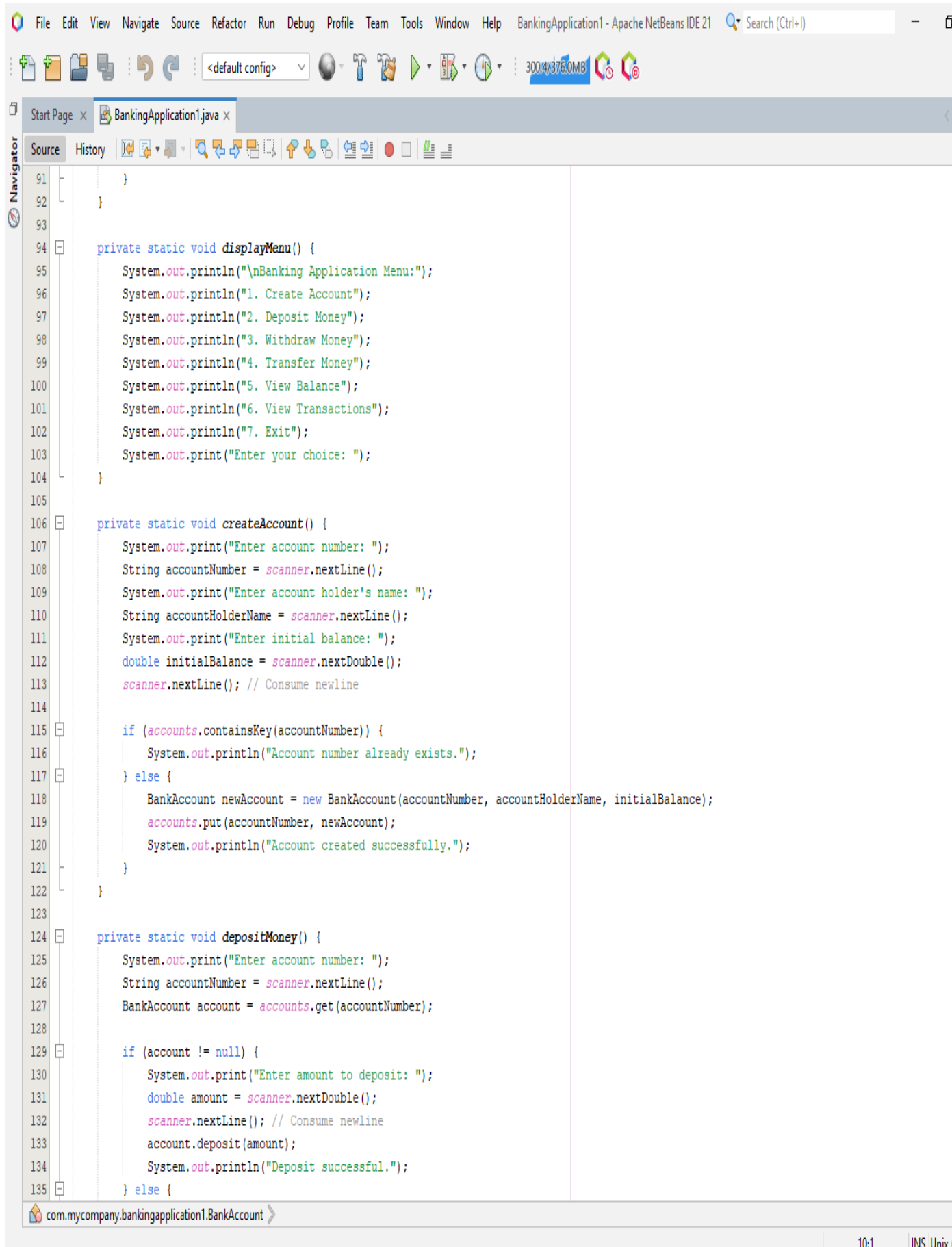
Start Page x BankingApplication1.java x

Source History

Navigator

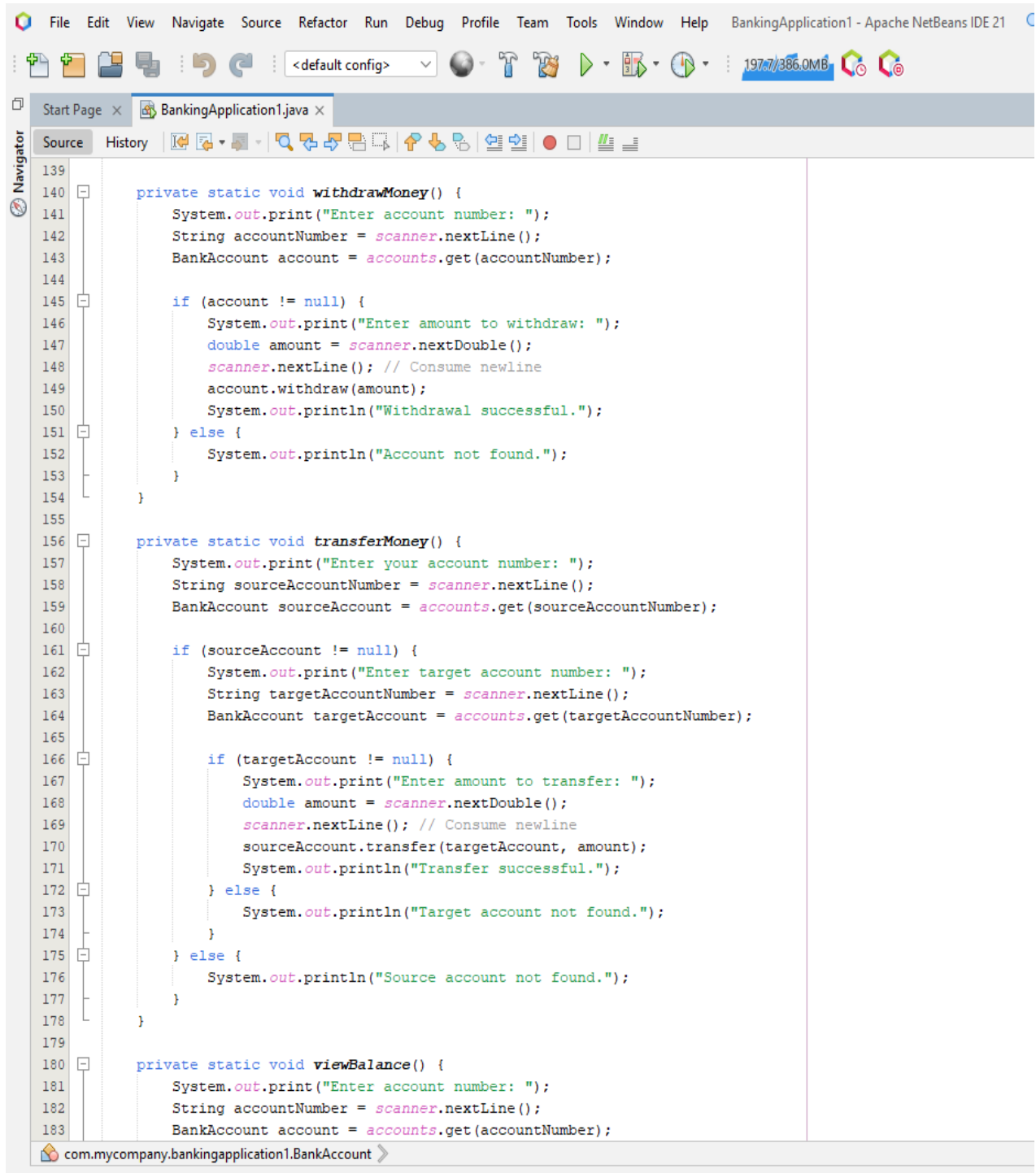
```
46
47 public void withdraw(double amount) {
48     if (amount > 0 && amount <= balance) {
49         balance -= amount;
50         transactions.add("Withdrew: " + amount);
51     } else {
52         System.out.println("Invalid withdrawal amount or insufficient balance.");
53     }
54 }
55
56 public void transfer(BankAccount targetAccount, double amount) {
57     if (amount > 0 && amount <= balance) {
58         this.withdraw(amount);
59         targetAccount.deposit(amount);
60         transactions.add("Transferred: " + amount + " to " + targetAccount.getAccountNumber());
61         targetAccount.getTransactions().add("Received: " + amount + " from " + this.getAccountNumber());
62     } else {
63         System.out.println("Invalid transfer amount or insufficient balance.");
64     }
65 }
66
67
68 public class BankingApplication1 {
69     private static final HashMap<String, BankAccount> accounts = new HashMap<>();
70     private static final Scanner scanner = new Scanner(System.in);
71
72     public static void main(String[] args) {
73         while (true) {
74             displayMenu();
75             int choice = scanner.nextInt();
76             scanner.nextLine(); // Consume newline
77
78             switch (choice) {
79                 case 1 -> createAccount();
80                 case 2 -> depositMoney();
81                 case 3 -> withdrawMoney();
82                 case 4 -> transferMoney();
83                 case 5 -> viewBalance();
84                 case 6 -> viewTransactions();
85                 case 7 -> {
86                     System.out.println("Exiting application.");
87                     return;
88                 }
89                 default -> System.out.println("Invalid choice. Please try again.");
90             }
91         }
92     }
93 }
```

com.mycompany.bankingapplication1.BankAccount >



```
91     }
92 }
93
94 private static void displayMenu() {
95     System.out.println("\nBanking Application Menu:");
96     System.out.println("1. Create Account");
97     System.out.println("2. Deposit Money");
98     System.out.println("3. Withdraw Money");
99     System.out.println("4. Transfer Money");
100    System.out.println("5. View Balance");
101    System.out.println("6. View Transactions");
102    System.out.println("7. Exit");
103    System.out.print("Enter your choice: ");
104 }
105
106 private static void createAccount() {
107     System.out.print("Enter account number: ");
108     String accountNumber = scanner.nextLine();
109     System.out.print("Enter account holder's name: ");
110     String accountHolderName = scanner.nextLine();
111     System.out.print("Enter initial balance: ");
112     double initialBalance = scanner.nextDouble();
113     scanner.nextLine(); // Consume newline
114
115     if (accounts.containsKey(accountNumber)) {
116         System.out.println("Account number already exists.");
117     } else {
118         BankAccount newAccount = new BankAccount(accountNumber, accountHolderName, initialBalance);
119         accounts.put(accountNumber, newAccount);
120         System.out.println("Account created successfully.");
121     }
122 }
123
124 private static void depositMoney() {
125     System.out.print("Enter account number: ");
126     String accountNumber = scanner.nextLine();
127     BankAccount account = accounts.get(accountNumber);
128
129     if (account != null) {
130         System.out.print("Enter amount to deposit: ");
131         double amount = scanner.nextDouble();
132         scanner.nextLine(); // Consume newline
133         account.deposit(amount);
134         System.out.println("Deposit successful.");
135     } else {
```

com.mycompany.bankingapplication1.BankAccount >



```
139
140 private static void withdrawMoney() {
141     System.out.print("Enter account number: ");
142     String accountNumber = scanner.nextLine();
143     BankAccount account = accounts.get(accountNumber);
144
145     if (account != null) {
146         System.out.print("Enter amount to withdraw: ");
147         double amount = scanner.nextDouble();
148         scanner.nextLine(); // Consume newline
149         account.withdraw(amount);
150         System.out.println("Withdrawal successful.");
151     } else {
152         System.out.println("Account not found.");
153     }
154 }
155
156 private static void transferMoney() {
157     System.out.print("Enter your account number: ");
158     String sourceAccountNumber = scanner.nextLine();
159     BankAccount sourceAccount = accounts.get(sourceAccountNumber);
160
161     if (sourceAccount != null) {
162         System.out.print("Enter target account number: ");
163         String targetAccountNumber = scanner.nextLine();
164         BankAccount targetAccount = accounts.get(targetAccountNumber);
165
166         if (targetAccount != null) {
167             System.out.print("Enter amount to transfer: ");
168             double amount = scanner.nextDouble();
169             scanner.nextLine(); // Consume newline
170             sourceAccount.transfer(targetAccount, amount);
171             System.out.println("Transfer successful.");
172         } else {
173             System.out.println("Target account not found.");
174         }
175     } else {
176         System.out.println("Source account not found.");
177     }
178 }
179
180 private static void viewBalance() {
181     System.out.print("Enter account number: ");
182     String accountNumber = scanner.nextLine();
183     BankAccount account = accounts.get(accountNumber);
184
185     com.mycompany.bankingapplication1.BankAccount >
```

BankingApplication1 - Apache NetBeans IDE 21

Search (Ctrl+I)

<default config>

293.9/386.0MB

Start Page x BankingApplication1.java x

Source History

```

178 }
179
180 private static void viewBalance() {
181     System.out.print("Enter account number: ");
182     String accountNumber = scanner.nextLine();
183     BankAccount account = accounts.get(accountNumber);
184
185     if (account != null) {
186         System.out.println("Current balance: " + account.getBalance());
187     } else {
188         System.out.println("Account not found.");
189     }
190 }
191
192 private static void viewTransactions() {
193     System.out.print("Enter account number: ");
194     String accountNumber = scanner.nextLine();
195     BankAccount account = accounts.get(accountNumber);
196
197     if (account != null) {
198         System.out.println("Transaction history:");
199         for (String transaction : account.getTransactions()) {
200             System.out.println(transaction);
201         }
202     } else {
203         System.out.println("Account not found.");
204     }
205 }
206 }
207

```

com.mycompany.bankingapplication1.BankAccount

Output - Run (BankingApplication1) x

Compiling 1 source file with javac [debug target 21] to target\classes

--- exec:3.1.0:exec (default-cli) @ BankingApplication1 ---

Banking Application Menu:

1. Create Account
2. Deposit Money
3. Withdraw Money
4. Transfer Money
5. View Balance
6. View Transactions
7. Exit

Enter your choice:

Run (BankingApplication1) 50%