Object Oriented Programing 1

Lecture 7: Decision making (using Loops), Break & Continue Statement in Java

By

Elubu Joseph

MSci.IS

Email: josebulinda@gmail.com

or

jose@kumiuniversity.ac.ug

Agenda

1. Decision Making using
    i. Loop methods
    ii. Break & Continue Statement

2. Sample Programing Project

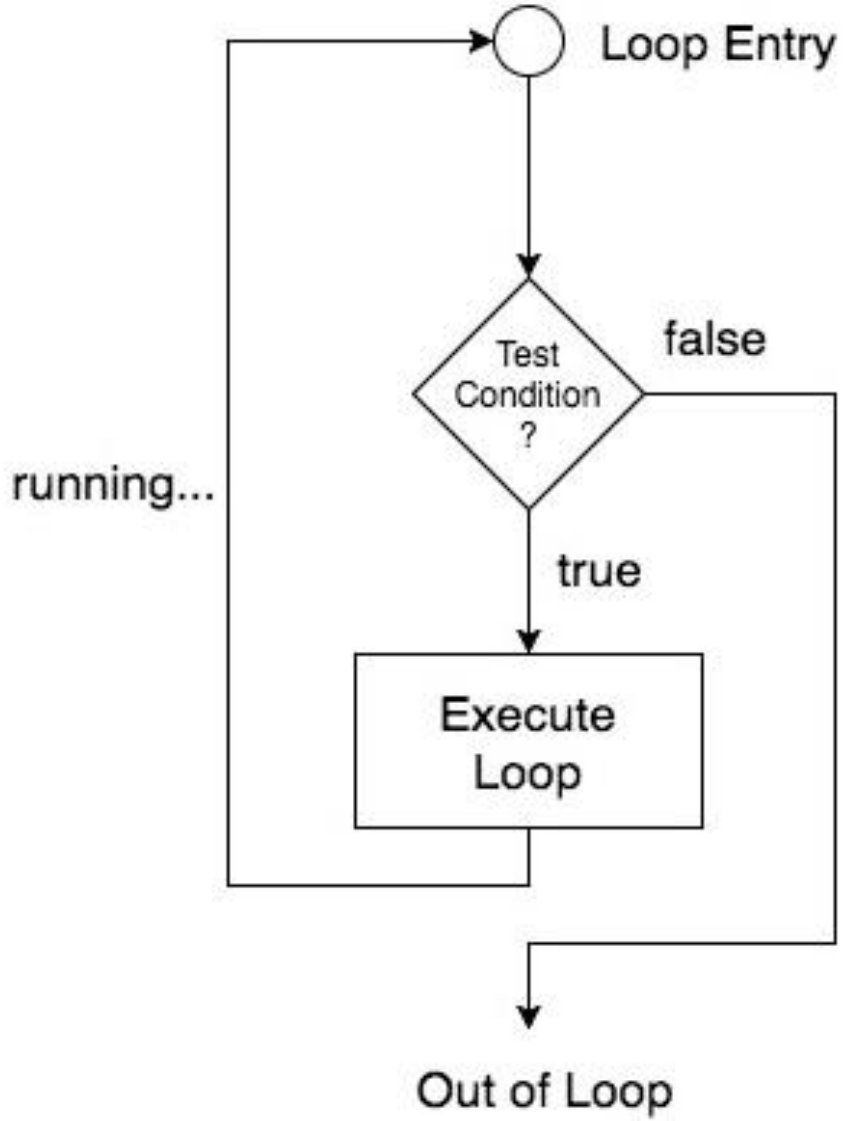
Recap on how to make decision using Loops in java

Recap on How to use Loops in Java

In any programming language including java, loops are used to execute a set of statements repeatedly until a particular condition is satisfied.

How it Works

The diagram below depicts a loop execution,



How Loops work explained

As per the above diagram, if the Test Condition is true, then the loop is executed, and if it is false then the execution breaks out of the loop. After the loop is successfully executed the execution again starts from the Loop entry and again checks for the Test condition, and this keeps on repeating.

The sequence of statements to be executed is kept inside the curly braces { } known as the **Loop body**. After every execution of the loop body, **condition** is verified, and if it is found to be **true** the loop body is executed again. When the condition check returns **false**, the loop body is not executed, and execution breaks out of the loop.

Types of Loop

There are 3 common types of Loop in java language, namely:

1. while loop
2. for loop
3. do while loop

Basic Structure of while loop

can be addressed as an **entry control** loop.

It is completed in 4 steps.

1. Variable initialization.(e.g int x = 0;)
2. condition(e.g while(x <= 10))
3. Statement to be executed
4. Variable increment or decrement

   (x++ or x-- or x = x + 2)

Syntax :

```
int x=4;
while(x<=10) {
System.out.println (“I love it\n”);
x++;
}
```

while loop

Example: Program to print first 9 natural numbers

```
static void whiler( ) {
     int x; x = 0;
     while(x < 10) {
          System.out.print(x);
          x++;
     }
}
```

Output

0 1 2 3 4 5 6 7 8 9

Jumping Out of Loops

Sometimes, while executing a loop, it becomes necessary to skip a part of the loop or to leave the loop as soon as certain condition becomes **true**. This is known as jumping out of loop. We use two word in this process, that is: -

1) break statement

2) Continue statement

1) Break statement

When break statement is encountered inside a loop, the loop is immediately exited and the program continues with the statement immediately following the loop.

```
while( condition check )
{
   statement-1;
   statement-2;
   if( some condition)
   {
      break;
   }
   statement-3;
   statement-4;
}
```

Jumps out of the loop, no matter how many cycles are left, loop is exited.

Sample Break code

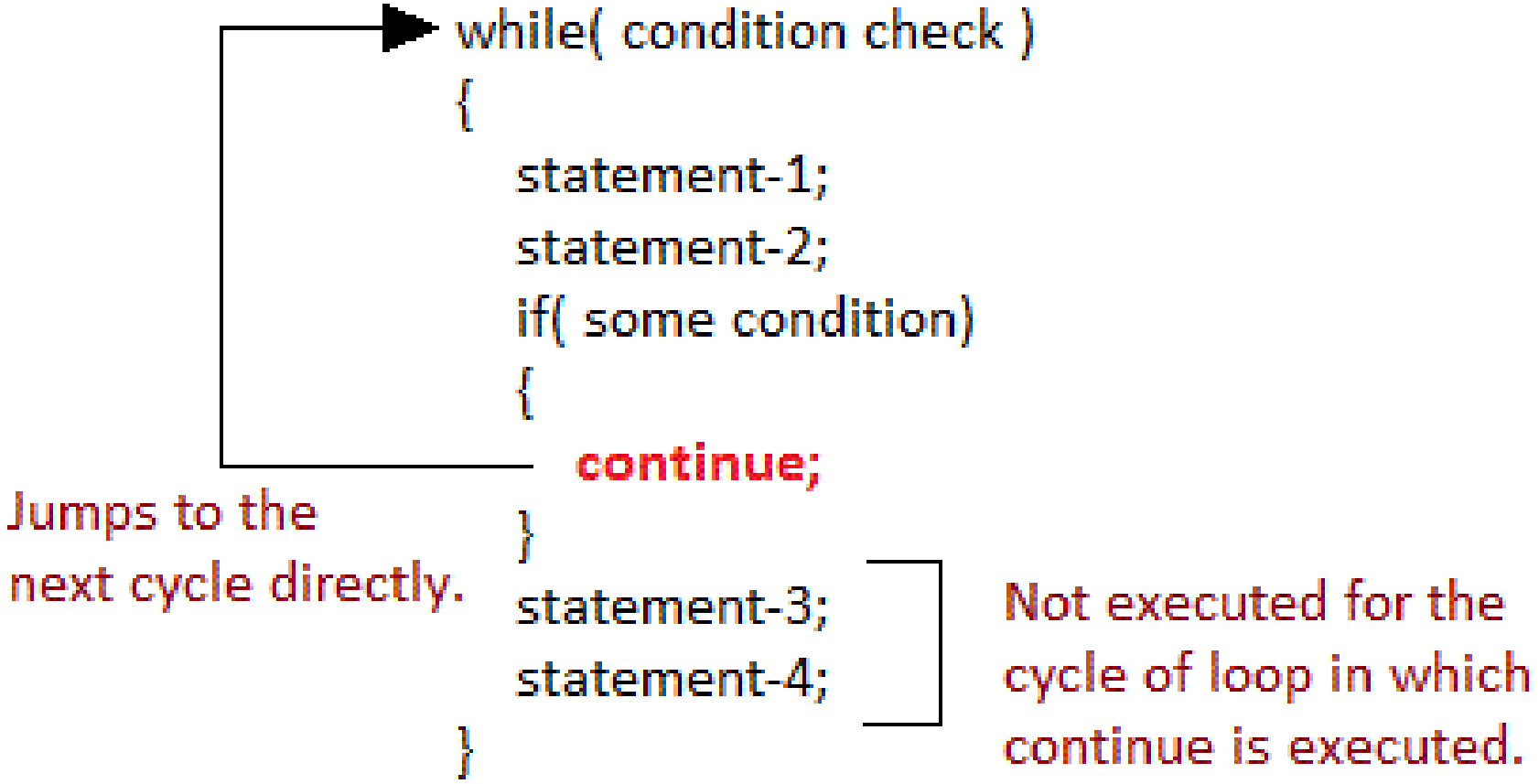
```
static void breaking(){
        int x; x = 1;
        while(x <=10){
                System.out.println( x);
         x++;
         if(x==7){
        break;
         }
        System.out.print("Now at ");
        }
         System.out.println("Now out of the loop ");
}
```

OUTPUT

```
1
Now at 2
Now at 3
Now at 4
Now at 5
Now at 6
Now out of the loop
```

2) continue statement

It causes the control to go directly to the test-condition and then continue the loop process. On encountering continue, cursor leave the current cycle of loop, and starts with the next cycle.



Sample Continue code

```
static void continuing(){
        int x; x = 1;
        while(x <=10){
                System.out.println(x);
         x++;
         if(x<=3){
           continue;
         }
         System.out.println("Now at ");
         }
          System.out.println("Now out of the loop ");
}
```

OUTPUT

```
1
2
3
Now at 4
Now at 5
Now at 6
Now at 7
Now at 8
Now at 9
Now at 10
Now at Now out of the loop
```

do while loop

do while loop Structure Vs while loop

do statement evaluates the body of the loop first and then checks the condition at the end, the condition is checked using while statement. It means that the body of the loop will be executed at least once, even though the starting condition inside while is initialized to be **false**. General syntax is,

```
do {
Statement block;
Increment statement;
}
while(condition);
```

Vs

```
while(condition){
Statement block;
Increment statement;
}
```

Basic Structure of do while loop

Do while loop is also completed in 4 steps.

1. Variable initialization.(e.g `int x = 0;`)
2. Statement to be executed
3. Variable increment or decrement
   (`x++` or `x--` or `x = x + 2`)
4. condition(e.g `while(x <= 10)`)

Syntax :

```
int x=3;
do{
System.out.println("I love it\n");
x++;
}
while(x<=10);
```

Output: I love it * 8

do loop

Example: Program to print first 5 natural numbers

```
static void dower( ) {
     int x; x = 1;
     do{
          System.out.print("\t"+x);
          x++;
     }
      while(x <= 5);
     System.out.println();
}
```

Output

1 2 3 4 5

Example: Program to print first 6 multiples of 5.

```
void dower1() {
     int a, i; a = 5; i = 1;
     do {
          System.out.print("\t“+ a*i);
          i++;
     }
     while(i <= 6);
      System.out.println();
}
```

OUTPUT:

5 10 15 20 25 30

for Loop

for loop Structure

for loop is used to execute a set of statements repeatedly until a particular condition is satisfied. for loop is an **open ended loop.**. General format is,

```
for(initialization; condition; increment/decrement) {

statement-block;

}
```

for loop Structure +

In for loop, we have exactly two semicolons, first one after initialization and second after the condition.

In this loop we can have more than one initialization or increment/decrement, separated using comma operator. But it can have only one **condition**.

The for loop is executed as follows:

1. It first evaluates the initialization code.
2. Then it checks the condition expression.
3. If it is **true**, it executes the for-loop body.
4. Then it evaluate the increment/decrement condition and again follows from step 2.
5. When the condition expression becomes **false**, it exits the loop.

Example: Program to print first 10 natural numbers

```
static void forla( ) {
     int x;
     for(x = 1; x <= 10; x++) {
     System.out.print(x);
     }
      System.out.println();
}
```

Output

1 2 3 4 5 6 7 8 9 10

Nested for loop

We can also have nested for loops, i.e one for loop inside another for loop. Basic syntax is,

```
for(initialization; condition; increment/decrement)
{
      for(initialization; condition; increment/decrement)
        {
       statement ;
      }
}
```

Example: Program to print half Pyramid of numbers

```
static void nestedForla( ) {
     int i, j;
     /* first for loop */
     for(i = 1; i < 5; i++) {
          System.out.println();
     /* second for loop inside the first */
          for(j = i; j > 0; j--) {
               System.out.print(j);
          }
      System.out.println();
     }
}
```

OUTPUT:

```
1
21
321
4321
```

Electronic Money Transaction Application programing Sample Project in Java

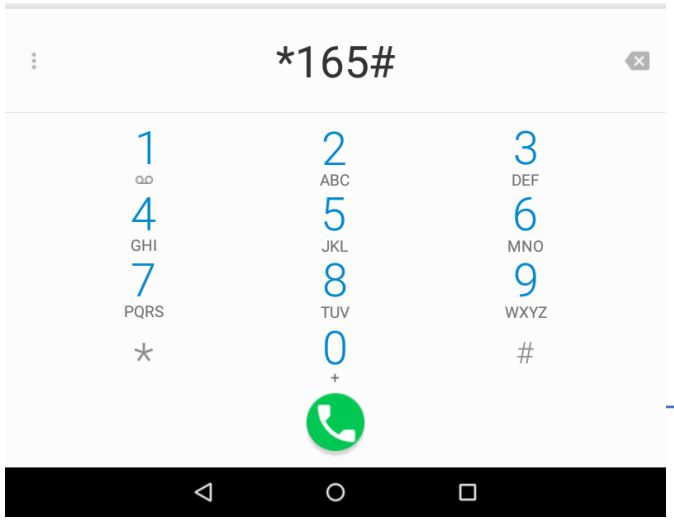
In a number of countries including Uganda, Mobile Electronic transactions is currently transforming the way people do transactions using their phones.

Mobile phones users will need to buy airtime, make various forms of payments, transfer money from their bank accounts to the mobile handsets etc.

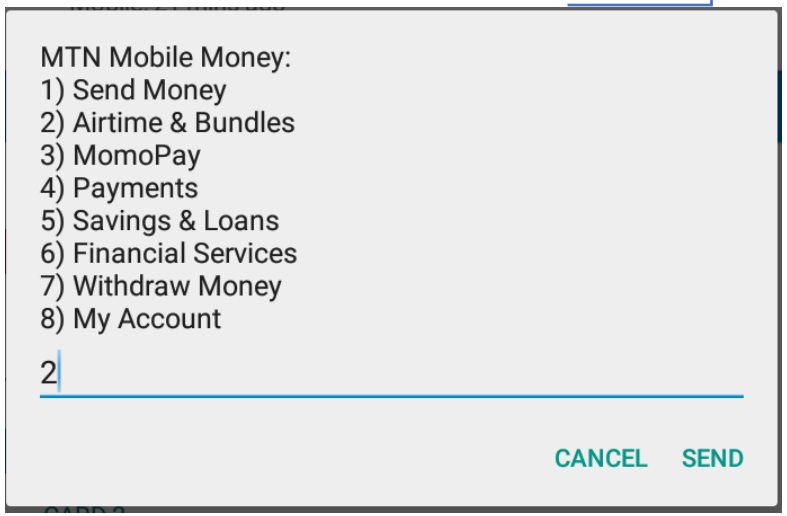
To put this into perspective, we will simulate MTN's Mobile application transaction process as seen below.

MTN Mobile Money Transaction Menu/steps

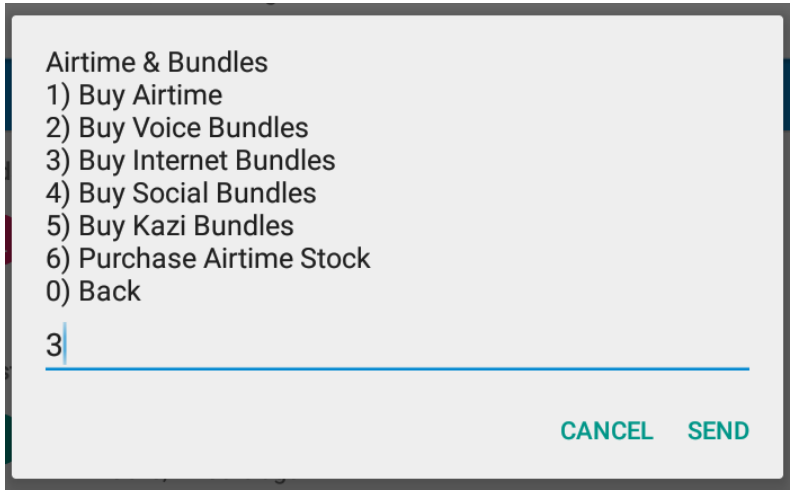
Lets think of a system that a allow a mobile user to buy electronic airtime/voice bundles. The user will have to start from some where and that is e.g. *165# for MTN Uganda & *185# for Airtel Uganda.



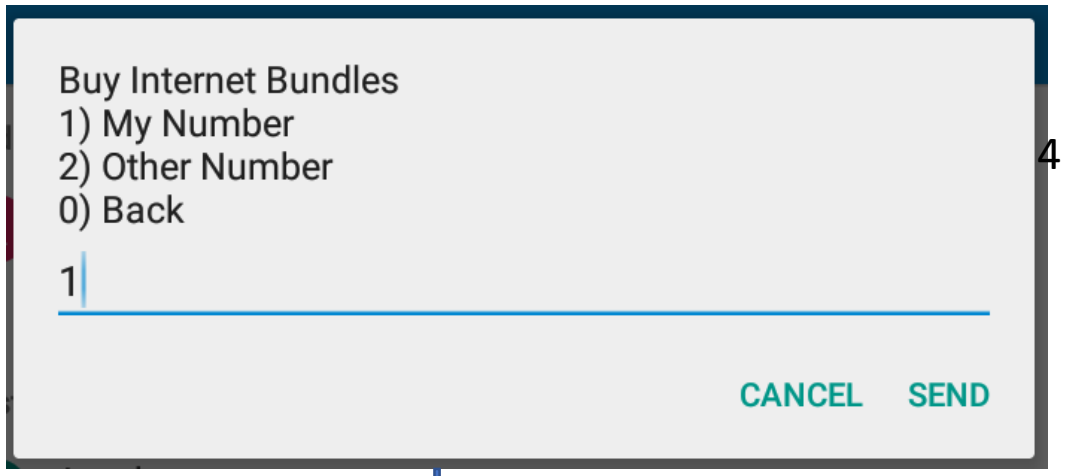
1



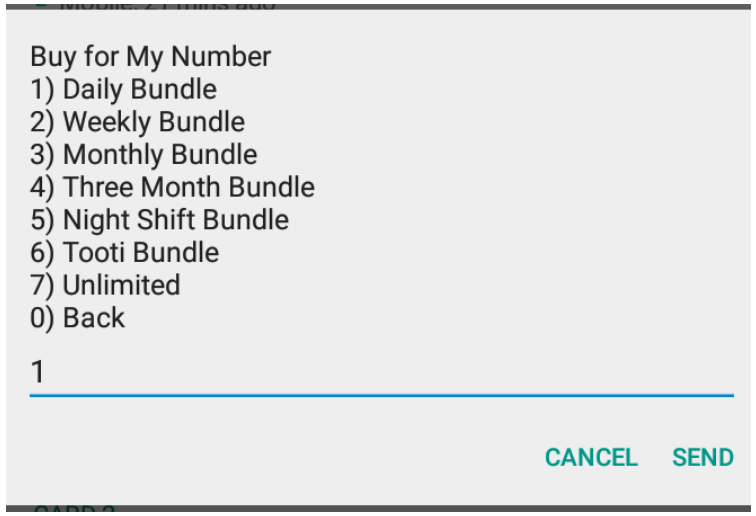
2



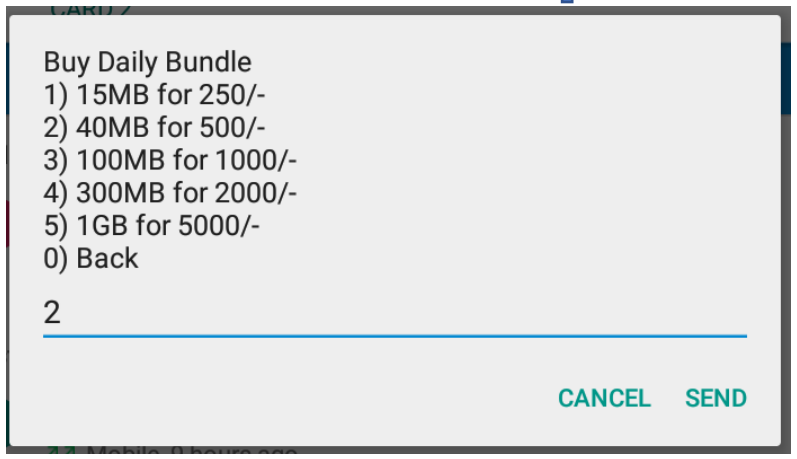
3


Buy Internet Bundles
1) My Number
2) Other Number
0) Back
1
CANCEL
SEND

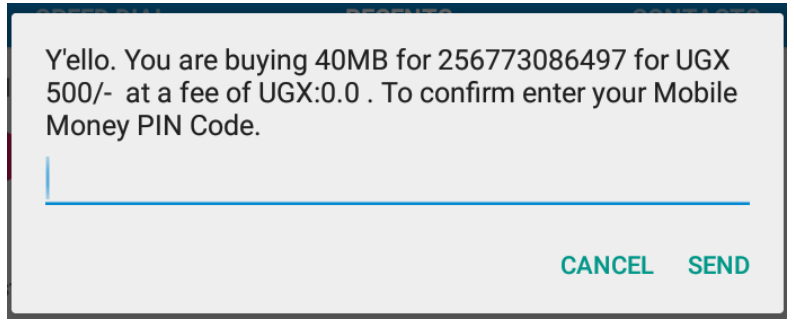
4


Buy for My Number
1) Daily Bundle
2) Weekly Bundle
3) Monthly Bundle
4) Three Month Bundle
5) Night Shift Bundle
6) Tooti Bundle
7) Unlimited
0) Back
1
CANCEL
SEND

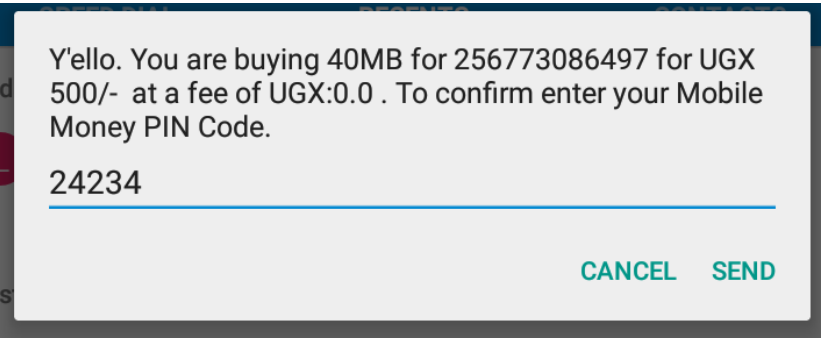
5


Buy Daily Bundle
1) 15MB for 250/-
2) 40MB for 500/-
3) 100MB for 1000/-
4) 300MB for 2000/-
5) 1GB for 5000/-
0) Back
2
CANCEL
SEND

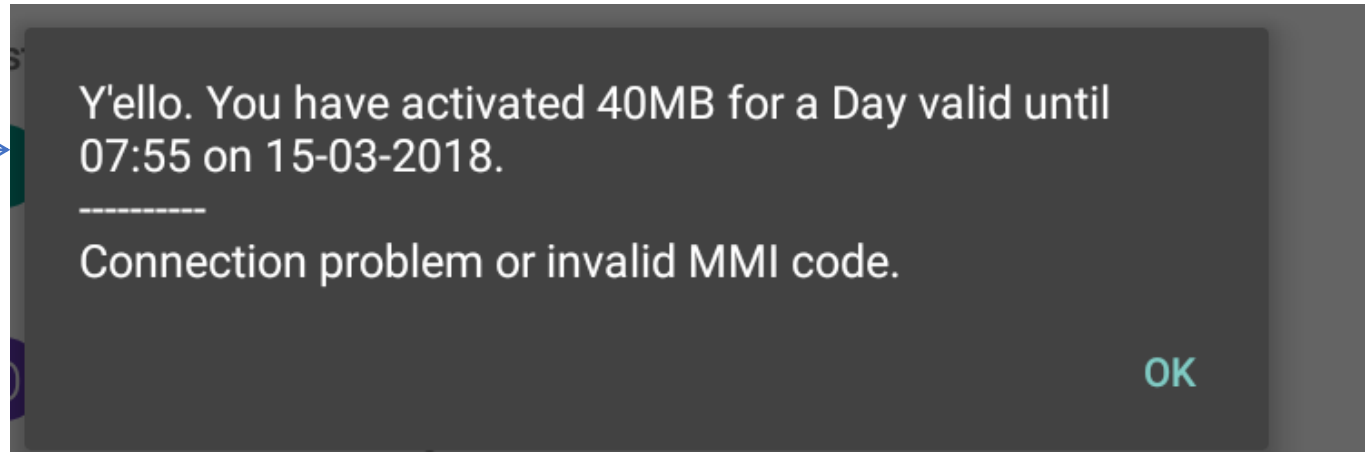
6


Y'ello. You are buying 40MB for 256773086497 for UGX 500/- at a fee of UGX:0.0 . To confirm enter your Mobile Money PIN Code.
CANCEL
SEND

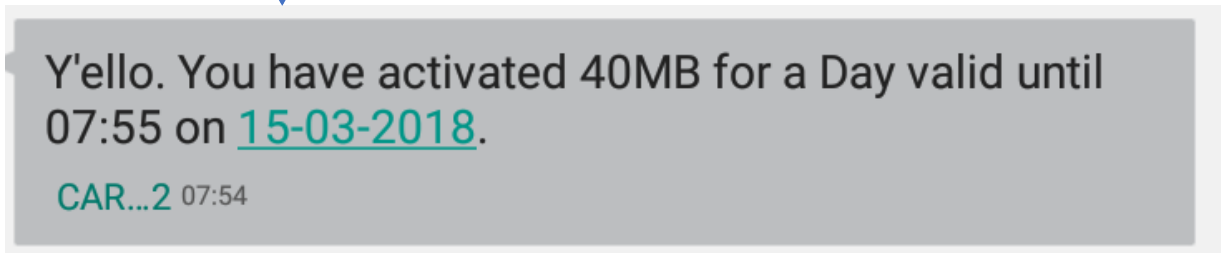
7



8



9 Final Stage



Sample program

This sample program has two classes: - **EMoney** and **Runner** which are both housed by **e-money** package.

Emoney class has main() method which runs the Runner class method called **startingPoint()** through the Runner class object(obj);

startingPoint() method runs the rest of the methods such as: -

1. airtimeBundles();

2. invalidError();

3. unknownError();

Located within the Runner class.

Lets Code now…….

```
package e.money;

public class EMoney {

    public static void main(String[] args) {
        Runner obj = new Runner();
        obj.startingPoint();
    }
}
```

Lets Code now…….

```
package e.money;

import java.util.Scanner;

public class Runner {
//Global Varriables declaration
     static int answer, num2=1;
     static String start;
     static int myNum=773086497;
     static int option=775614411, otherNum, myPIN=6677, PIN;
     static double amount;
```

Lets Code now…….+

```
public static void startingPoint(){
       Scanner ob = new Scanner(System.in);
        System.out.println("\nTo Access MTN Service Dial *165# &  press Enter Key
");
        start = ob.nextLine();

        if(start.equals("*165#")){
            System.out.println(" Welcome to MTN Mobile Money\n ---------------\n");
            System.out.println(" 1. Send Money.\n 2. Airtime & Bundles\n 3.
MomoPay\n 4. Payments\n 5. Savings & Loans\n 6. Financial Services\n 7. Withdraw
Money\n 8. My Account\n");
            answer = ob.nextInt();
            if(answer==1){
             System.out.println(" Send Money\n----------------------------\n");

            }
```

Lets Code now…….++

```
else if(answer==2){
                airtimeBundles();
            }
            else if(answer==3){
                System.out.println(" MOMo Pay\n-----------------\n");
            }
            else{
               unknownError();
            }
        }
        else{
           unknownError();
        }
    }
```

Lets Code now…….

```
//Other methods
    public static void airtimeBundles(){
        System.out.println(" Buy Airtime and Voice Bundles\n----\n");
         System.out.println(" 1. Airtime\n 2. Voice BBundles");
         Scanner ob = new Scanner(System.in);
         answer = ob.nextInt();
         if(answer == 1){
             System.out.println(" Airtime\n--------------");
             System.out.println(" 1. For self\n 2. Other number");
             answer = ob.nextInt();
                 if(answer == 1){
                  System.out.println(" For self \n Enter Amount\n");
                  amount = ob.nextDouble();
```

```
System.out.println(" You're buying Airtime worth UGX"+amount +" to "+myNum);
                System.out.print("Enter PIN to prosceed: ");
                PIN = ob.nextInt();
                   if(PIN!=myPIN){
                    while(PIN!=myPIN){
                        System.out.println(“incorrect PIN Try again ");
                        PIN = ob.nextInt();
                        num2++;
                         if(num2==3){
                            System.out.println("You entered wrong PIN Three“+
times\nTry again after 2 Minutes");
                            break;
                        }
                    }
                   System.out.println("Tansaction Successful.");
                }
                else{

                    System.out.println("Tansaction Successful.");
                }
            }
```

Lets Code now –buy for other number

```
else if(answer == 2){
                System.out.println(" Other number\n---------------");
                System.out.println(" Enter Number\n");
                otherNum = ob.nextInt();
                System.out.println(" For self \n Enter Amount\n");
                amount = ob.nextDouble();
                if(otherNum == option){
                    System.out.println(" You're buying Airtime worth UXG "+amount
+" for Abraham ");
                }
                else{
                   System.out.println(" You're buying Airtime worth UGX "+amount +"
to "+otherNum);
                }

                System.out.print("Enter PIN to prosceed: ");
                PIN = ob.nextInt();
```

Lets Code now…….

```
if(PIN!=myPIN){
        while(PIN!=myPIN){
                System.out.println("You entered invalid PIN Please Try again ");
                     PIN = ob.nextInt();
                     num2++;
                  }
                System.out.println("Tansaction Successful.");
              }
              else{
                System.out.println("Tansaction Successful.");
              }
            }
            else {
                  invalidError();
                  }
        }
```

Lets Code now…….

```
else if(answer == 2){
    System.out.println(" Voice Bundles\n--------");
        }
   else{

            invalidError();
        }
   }
```

Lets Code now…….Error Methods

```
//Error
    static void invalidError(){
        System.out.println("Invalid entry");
    }
    static void  unknownError(){
         System.out.println(" Unknown Application ");
    }
}
```

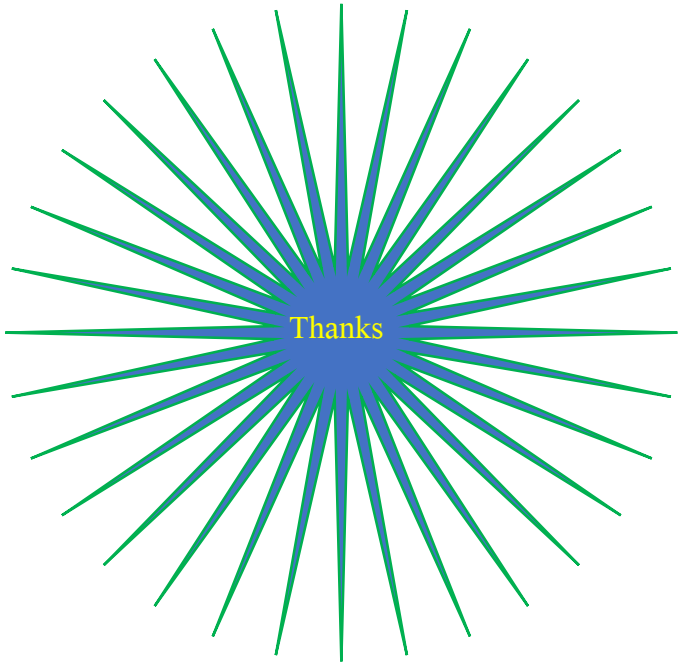
Work the rest of the needed code

NOTE that we have only dealt with the purchase of Airtime to self and other number. So you will build the rest of the sections as displayed in the main menu.

Summary

1. Recap on Decision Making
    i. Loop methods –introduced three loop functions dealt with while loop(layout and sample program)
    ii. Had Electronic money transaction programing Sample project.

Thank you for Listening



Reference

Java IF ... else. (n.d.). Retrieved April 11, 2022, from https://www.w3schools.com/java/java_conditions.asp

University of Helsinki. (n.d.). *Conditional statements and conditional operation*. Conditional statements and conditional operation - Java Programming. Retrieved April 11, 2022, from https://java-programming.mooc.fi/part-1/6-conditional-statements

Java IF...else statement. Programiz. (n.d.). Retrieved April 16, 2022, from https://www.programiz.com/java-programming/if-else-statement

Java loops. Studytonight.com. (n.d.). Retrieved April 18, 2022, from https://www.studytonight.com/java/loops-in-java.php