CSC207 · INTRODUCTION TO ALGORITHMS · CA 2024

CSC207
CA 2024

Every question on the CSC207 CA 2024 paper (20/11/2024), word for word, solved in full in the course pseudocode: reverseList, arrays and files, exact-output tracing with continue and break, and the auto teller algorithm with its flowchart.

Section A and B | 8 worked solutions | Every output checked

Solved by **Tabe Rickson** · taberickson.com/csc207/ca2024
Based on the CSC207 Introduction to Algorithms CA 2024 paper.

What is inside

SECTION A · ARRAYS, FILES AND EXACT OUTPUT

A1a reverseList()
A1b Disadvantage of an array
A1c Files over arrays
A2a Exact output: switch
A2b Exact output: for loop
A2c Exact output: while with %
A2d Exact output: continue and break

SECTION B · THE AUTO TELLER: ALGORITHM AND FLOWCHART

B1 The auto teller machine

HOW TO USE THIS DOCUMENT

The **blue box** is the question exactly as it is on the paper. The solution follows it directly. `read(x)` reads a value, `println(...)` prints and ends the line, `print(...)` and `printf(...)` stay on the same line.

SECTION A

Arrays, files and exact output

A1a reverseList()

QUESTION A1A

Section A: Answer on a separate sheet. Answer all Questions for a Maximum of 30 marks.

1. Arrays are a relevant data structure in manipulating data. Consider an integer list data structure constructed with an array data type with the form:

```
struct ArrayList {
    int elements[MAX];
    int size;
};
```

a. Propose an algorithm called reverseList() which displays the values in an array called ArrayList[ ] in the reversed order. For example: the list 2, 3, 10, 23, 50 will be reversed to display: 50, 23, 10, 3, 2.

SOLUTION

```
void reverseList(ArrayList *list) {
    for (int i = list->size - 1; i >= 0; i--) {
        print(list->elements[i]);    // last element first
        if (i > 0)
            print(", ");              // no comma after the last value
    }
    println();
}
```

Start at the last position, `size - 1`, and walk back to 0. For 2, 3, 10, 23, 50 it displays `50, 23, 10, 3, 2`. The list itself is not changed: it is only displayed backwards.

A1b Disadvantage of an array

QUESTION A1B

b. What is the disadvantage of using an array in general?

SOLUTION

An array has a **fixed size**: `MAX` is chosen before the program runs. It cannot grow when more values arrive, and the unused places still take memory. Inserting or deleting in the middle also means shifting every element after it.

A1c Files over arrays

QUESTION A1C

c. What is the advantage of using file data structure over arrays?

(5 + 1 + 1 = 7 marks)

SOLUTION

A file is **permanent**: its data is stored on disk and is still there after the program ends or the computer is switched off. An array lives in memory and is lost when the program stops. A file can also hold far more data than an array, since it is not limited to a fixed `MAX`.

A2a Exact output: switch

QUESTION A2A

2. Carefully study the algorithms below and write down the "exact" output. Put your final output for each subquestion in a box.

a).

```
today=3;
switch(today)
{
    case 1: println("stay home");break;
    case 2: println("programing");break;
    case 3: println("mathematics");break;
    default: println(today);
}
```

SOLUTION

`today` is 3, so `case 3` runs, prints, and `break` leaves the switch.

EXACT OUTPUT

```
mathematics
```

A2b Exact output: for loop

QUESTION A2B

b).

```
i = 1;
for(j=1; j<10; j=j+1){
    println(i);
}
println();
```

SOLUTION

`j` takes 1, 2, ... 9: nine rounds, each prints `i`, which stays 1. The last `println()` has nothing to print, so it only ends the line: an empty line.

EXACT OUTPUT

```
1
1
1
1
1
1
1
1
1
```

Nine lines of `1`, then one empty line.

A2c Exact output: while with %

QUESTION A2C

c).

```
n=8;
i=1;
while(i<=n){
    if(i%2 ==0){
        printf(i);
    }
    i++;
}
```

SOLUTION

`i` runs from 1 to 8 and only the even numbers are printed: 2, 4, 6, 8. `printf` does not end the line, so they come out side by side with nothing between them.

EXACT OUTPUT

```
2468
```

A2d Exact output: continue and break

QUESTION A2D

d).

```
for(k=1; k<20; k+=4){
    if(k%4==0){
        continue;
    }
    if(k==17){
        break;
    }
    println(k);
    k++;
}
println(k);
```

(2 * 4 = 8 marks)

SOLUTION

Careful: `k` goes up by 1 inside the loop (`k++`) and then by 4 in the `for`.

k at the test	k%4 == 0?	k == 17?	Prints	Next k
1	no	no	1	1 + 1 + 4 = 6
6	no	no	6	6 + 1 + 4 = 11
11	no	no	11	11 + 1 + 4 = 16
16	yes: `continue`	skipped	nothing	16 + 4 = 20
20	20 < 20 is false: the loop ends			

`k` never equals 17, so `break` never runs. After the loop, `println(k)` prints 20.

EXACT OUTPUT

```
1
6
11
20
```

SECTION B

The auto teller: algorithm and flowchart

B1 The auto teller machine

QUESTION B1

Section B: Answer on a separate sheet (16 marks)

An automatic teller machine has a console as shown in the diagram below. The teller machine follows this sequence to assist a customer:

1. The customer will insert their card and enter a PIN. A customer is allowed at most three tries at their PIN. If they get it wrong three times the whole process ends without ejecting the card.
2. If the PIN is correct they will then select an action button (withdraw, deposit or balance).
3. Next they will select the account type (savings or cheque).
4. Finally, they will enter the amount in whole dollars (if appropriate), and press OK to confirm it, and the transaction will be processed.

The auto teller will eject the customer's card and stop the process if the customer presses the 'Cancel' button.

1	2	3	4	**Withdrawal**	**Savings**	
5	6	7	8	**Deposit**	**Cheque**	
9	0			**Balance**	**OK**	**Cancel**

Auto Teller Console

Using the different flow construct and subprogram, describe an algorithm which the auto teller could use to accept the details from the customer and act on them. Draw the corresponding flow chart.

SOLUTION

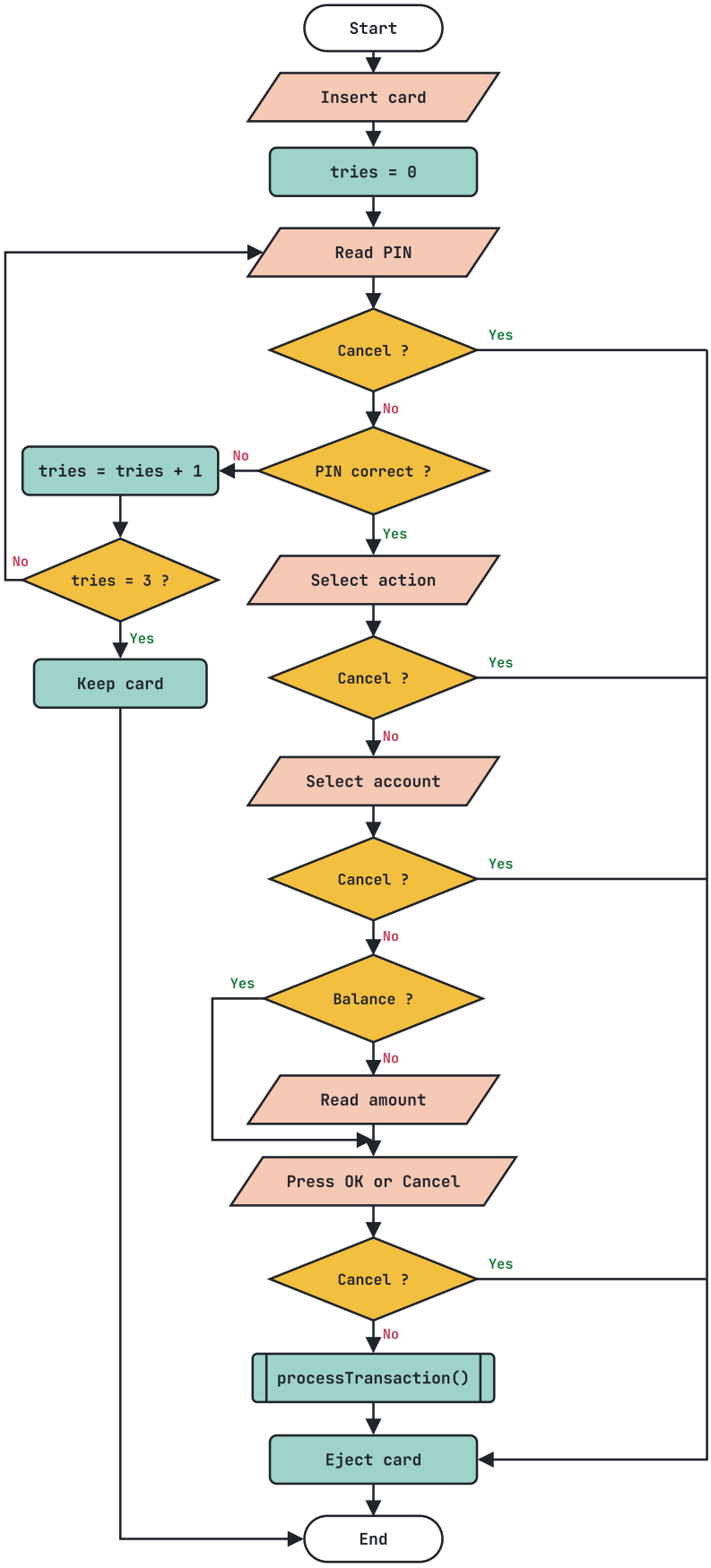
Algorithm

It uses all three flow constructs: **sequence** (the steps in order), **selection** (`if` and `switch`), **iteration** (the `while` loop for the three PIN tries), and **subprograms** for each job the machine does.

```
// Subprograms (one job each)
int  readPin();            // 4 digits from the keypad, or -1 if Cancel is pressed
bool checkPin(int pin);    // true if the PIN matches the card
int  readAction();         // 1 Withdrawal, 2 Deposit, 3 Balance, -1 Cancel
int  readAccount();        // 1 Savings, 2 Cheque, -1 Cancel
int  readAmount();         // whole dollars from the keypad
bool pressedOK();          // true for OK, false for Cancel
void ejectCard();
void keepCard();

void autoTeller() {
    int pin, action, account;
```

```
    int amount = 0;
    int tries = 0;
    bool correct = false;

    // 1. Card inserted: at most three tries at the PIN
    while (tries < 3 && correct == false) {
        pin = readPin();
        if (pin == -1) {             // Cancel
            ejectCard();
            return;
        }
        correct = checkPin(pin);
        if (correct == false)
            tries++;
    }
    if (correct == false) {          // wrong three times
        keepCard();                  // end without ejecting the card
        return;
    }

    // 2. Action button
    action = readAction();
    if (action == -1) {
        ejectCard();
        return;
    }

    // 3. Account type
    account = readAccount();
    if (account == -1) {
        ejectCard();
        return;
    }

    // 4. Amount (not needed for Balance), then OK
    if (action != 3)
        amount = readAmount();
    if (pressedOK() == false) {      // Cancel
        ejectCard();
        return;
    }
    processTransaction(action, account, amount);
    ejectCard();
}

void processTransaction(int action, int account, int amount) {
    switch (action) {
        case 1: withdraw(account, amount); break;
        case 2: deposit(account, amount); break;
        case 3: println("Balance: ", balance(account)); break;
    }
}
```

Flowchart



Flowchart for Section B: the auto teller