CEMC.UWATERLOO.CA | The CENTRE for EDUCATION in MATHEMATICS and COMPUTING

Grade 6 Math Circles

March 21/22/23, 2023

Control Flow - Solutions

Exercise Solutions

Exercise 1

Write a `for` loop that prints the numbers from 1 to 20. Then write the same thing but using a `while` loop.

Solution

For a `for` loop, we have the following code.

```
for(i=1; i<=20; i++) {
    print(i)
}
```

For a `while` loop, we just make our own counter! We have the following code.

```
i=0
while(i<20) {
    i++
    print(i)
}
```

Notice that the initial value for `i` and the conditions are slightly different for both loops. But they both do the same thing since they will both run 20 times. Both loops increment `i` once per loop, but the `while` loop has to do it within the body of the loop instead of in the first line.

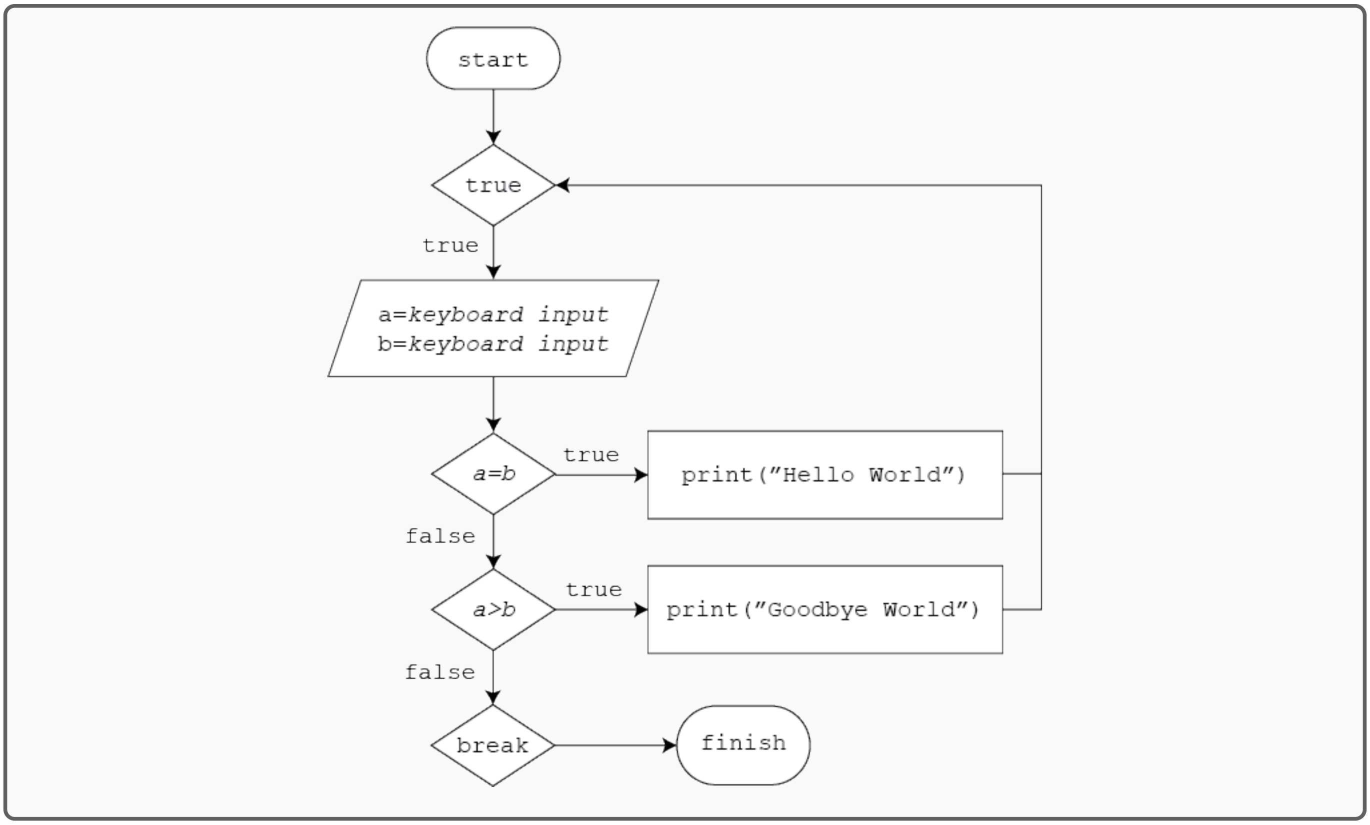
Exercise 2

Given the following code, draw a control flow diagram.

```
while(true) {
    a=keyboard input
    b=keyboard input
    if(a=b) {
        print("Hello world")
    } else if(a>b) {
        print("Goodbye world")
    } else {
        break
    }
}
```

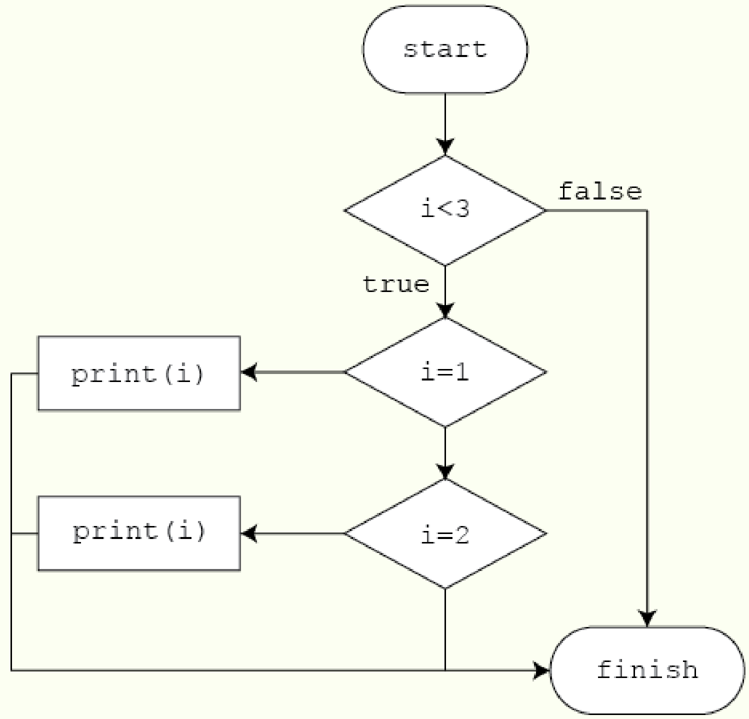
Solution

Even though the condition of the `while` loop is `true`, it is still a condition that will be included in the diagram. But since the condition is always `true`, we don't necessarily need to include a `false` option. Since `a` and `b` both take keyboard inputs, they will need an input symbol. We can combine both of these into one symbol. Our if statement has two conditions (the initial `if` statement and then the `else if` statement), so we need to reflect that in our diagram as well.



Exercise 3

Given the following control flow diagram, write code that matches it.



Solution

The first condition is whether `i` is less than 3. Since there is a `true` and `false` label, it is an `if` statement and not a `switch case` statement. It also isn't a `for` or `while` statement because we can see that the arrows don't flow in a loop (the arrows only head towards `finish`). There does not need to be an `else if` or `else` statement, because there are no other conditions to consider. Then the following conditions (`i=1` and `i=2`) must be `switch case` statements since there are no `true` or `false` labels. So we get the following code.

```
if(i<3) {
    switch(i) {
        case 1:
            print(i)
        case 2:
            print(i)
        default:

    }
}
```

Exercise 4

Consider the following algorithm.

1. The user inputs any given natural number (positive whole number), which we will call `n`.
2. If `n` is even, then divide it by 2. If `n` is odd, then multiply it by 3 and then add 1.
3. Repeat step 2 until `n` is one.
4. Output the number of steps the number, `n` takes to reach 1.

Write code for this algorithm and draw a control flow diagram to match your code (or vice versa). Use `is_even(n)` as the condition for whether `n` is even.

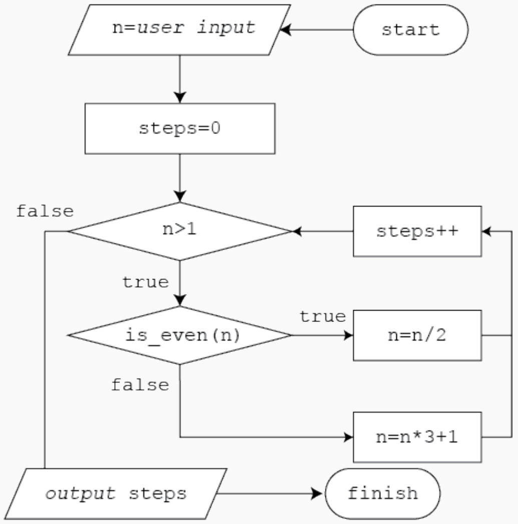
Solution

For this solution, we will use a `while` loop. We want to use our counter variable, which we will call `steps` at 0, because if `n` is already 0, the loop will not be entered, and it should have taken 0 steps to go from 1 to 1.

Note that in order to output `steps`, we need to use a `while` loop since `steps` would be a local variable to the `for` loop. Don't worry about this, since it's not something we are touching on.

Within the loop, we have two conditions; whether `n` is odd or even. But notice that since a number must be either odd or even, we can have an even condition and an `else` condition as the odd condition. So we have the following code and flow diagram.

```
n=user input
steps=0

while(n>1) {
    if(is_even(n)) {
        n = n/2
    } else {
        n = 3*n+1
    }
    steps++
}

output steps
```



Problem Set Solutions

1. What is the difference between an `if` statement and a `switch case` statement?

Solution
An `if` statement uses conditions that can compare multiple variables with each other, whereas a `switch case` has an `expression` that is only compared to different values it can take on, which we call cases.

Note that you can always model a `switch case` statement with an `if` statement, but not the other way around.

2. What is the difference between a `for` loop and a `while` loop?

Solution
A `for` loop has a built in counter, which is useful for writing loops that requires a counter, while a `while` loop does not.

Note that you can always add a manual counter to a `while` loop to model a `for` loop.

3. Circle whether or not the following pieces of code will stop looping (terminate). Give a brief explanation as to why or why not.

   (a) Terminates / Does not terminate

```
for(i=0; i<5; i++) {
    print(i)
}
```

   (b) Terminates / Does not terminate

```
for(i=0; i<5; i--) {
    print(i)
}
```

   (c) Terminates / Does not terminate

```
i=1
while(not i=0) {
    i=i/2
}
```

Solution

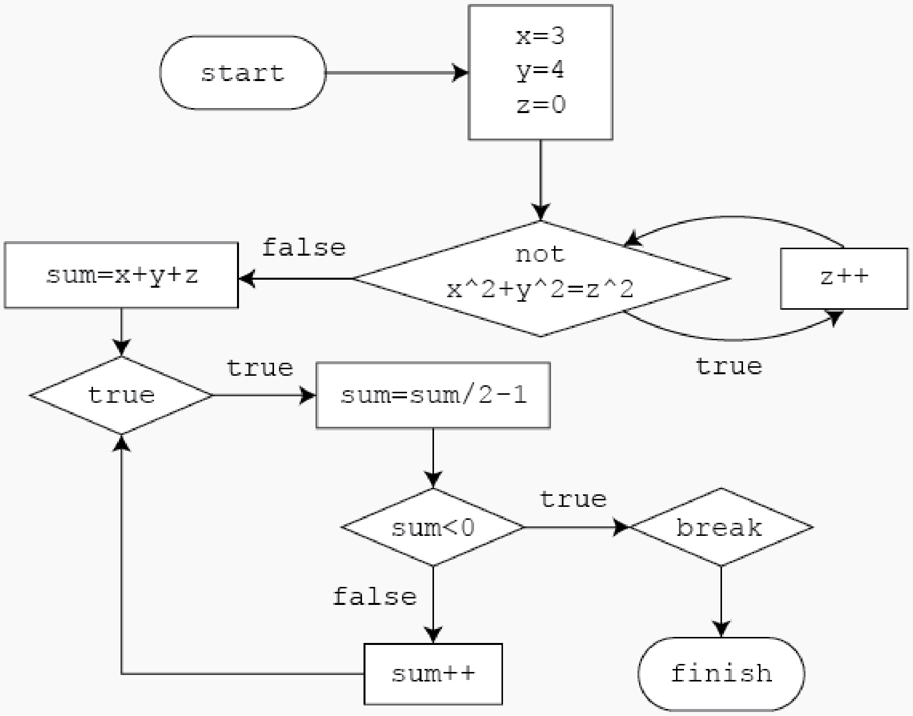
(a) Terminates (b) Does not terminate (c) Does not terminate

4. Based on the code, draw a control flow diagram. Note that `x^2` means $x^2 = x \times x$.

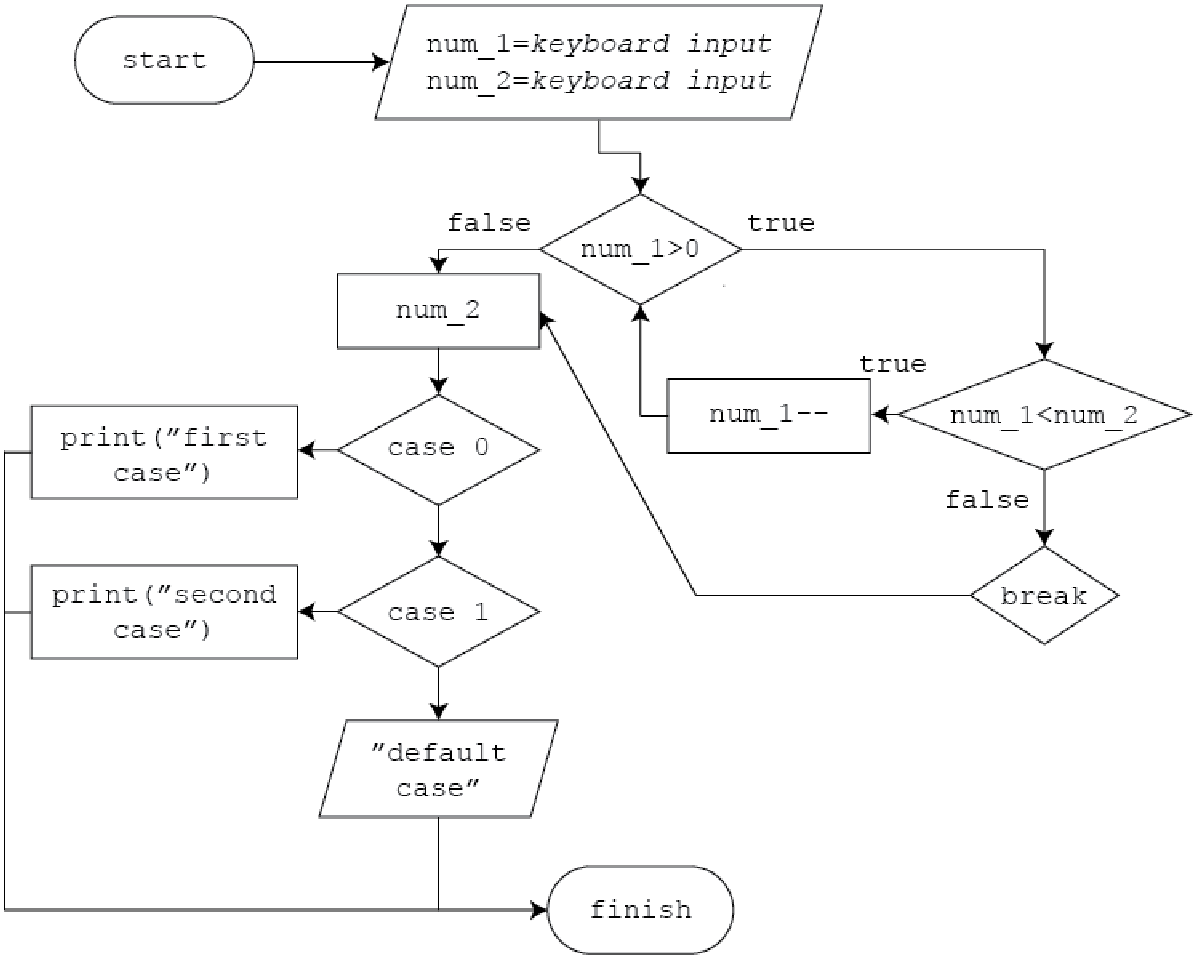
```
x=3
y=4
z=0

while(not x^2+y^2=z^2) {
    z++
}

sum=x+y+z

while(true) {
    sum=sum/2-1
    if(sum<0) {
        break
    } else {
        sum++
    }
}
```

Solution

Since the `for` loop has an empty body, the loop simply increases `z` once each loop. Also, since the `while` loop has `true` as its condition, we do not need to draw a `false` arrow out of the `while` loop condition. So, we have the following diagram.



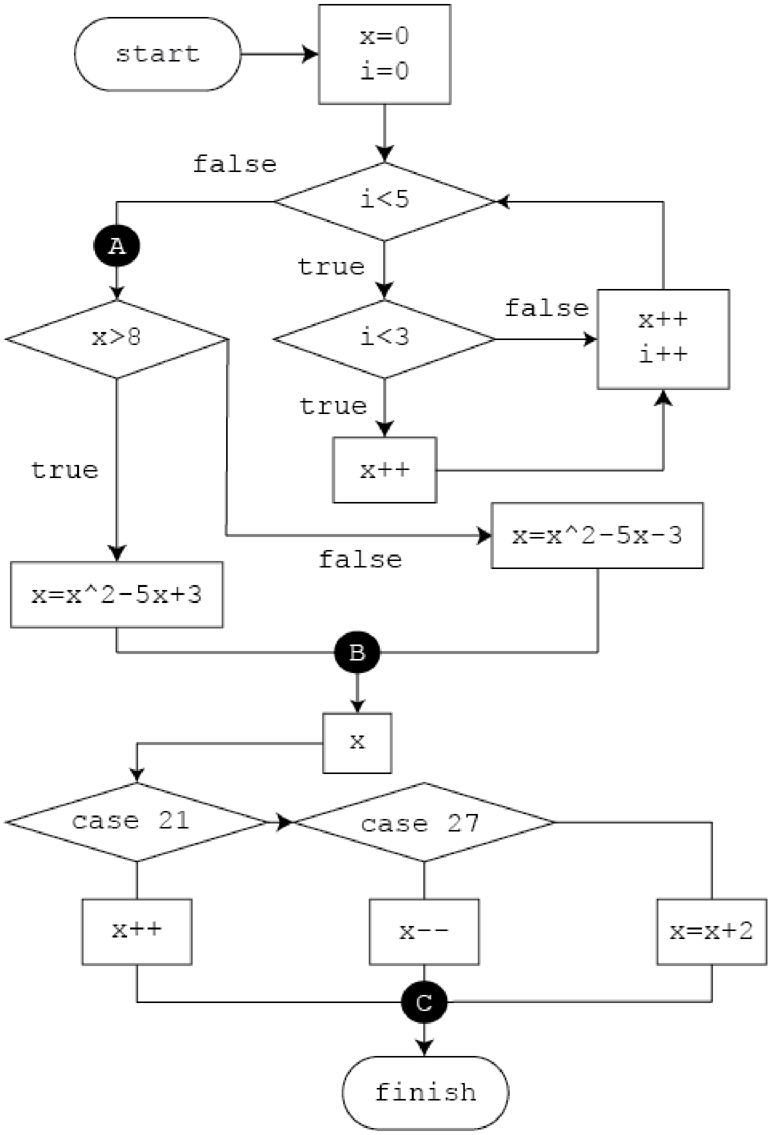
5. Based on the following diagram, write pseudo-code that would match this diagram.



Solution

```
num_1=keyboard input
num_2=keyboard input

while(num_1>0) {
    if(num_1<num_2) {
        num_1--
    } else {
        break
    }
}

switch(num_2) {
    case 0:
        print("first case")
    case 1:
        print("second case")
    default:
        output "default case"
}
```

6. Based on the following diagram, track the value of `x` at the given points (`A`, `B`, and `C`).



Solution

Let us first consider breaking this diagram into three sections: the `for` loop, the `if` statement, and a `switch case` statement.

Before the loop, we see that `x` starts at an initial value of `0`. In the `for` loop, `x` increases once every loop for 5 times, since the loop will loop 5 times from `i=0` to `i=4` before exiting. So `x` also increases 5 times (up to 5). Additionally, the code enters the `if` statement within the loop 3 times. So `x` increases another 3 times, and we get that `x=8` at point `A`.

Then since `x=8`, the `if` statement is false, and we calculate:

```
x => x^2-5x-3
  => 8^2-5(8)-3
  => 64-40-3
  => 21
```

So we get that `x=21` at point `B`

Finally, we look at the `switch case` statement. Since `x=21`, the first case is true and increase `x` by one again. So we get that `x=22` at point `C`.