Sample Assignment 1, 2020-09-09

Question 1 (Memory Content). Consider the following variable assignments:

```
long long int a = -6;
// Octal starts with "0"
long b = 0347;
// TAB character
char c = '\t';
```

Draw the memory content of these variables in *hexadecimal notation*. Ensure that the hex notation length equals their actual length. You may need to add leading zeroes.

Variable	Hex value
a	
b	
c	

Question 2 (Left and Right Shifts). Write the output (and the content of variables a,b,c in hexadecimal notation), after this snipped is executed:

```
1 int a = 47;
2 int b = -13;
3 char c = 'c';
4 // bitwise AND
5 cout << (a & b) << endl;
6 // bitwise OR
7 cout << (a | b) << endl;
8 // bitwise NOT
9 cout << (~b) << endl;
```

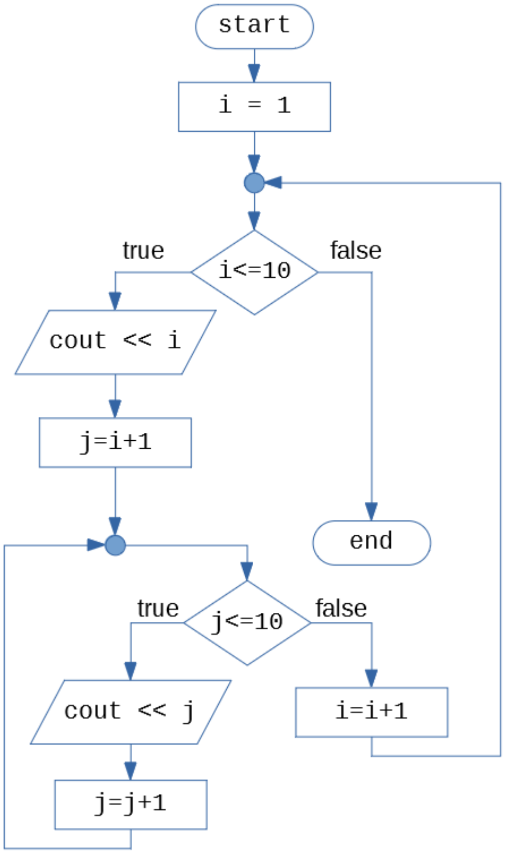
Variable	Hex value
a	
b	
c	
Line 5	
Line 7	
Line 9	

Question 3 (For-loops). Consider a regular ”for” loop like this:

```
for (int i=0; i<100; i++) { /*...*/ }
```

1. Is it legal to change the loop variable i in the body of the loop?
2. Is it legal to use the value i after the loop has finished?
3. Can we omit the any of the three parts in the for-loop? Can we omit all 3 parts as in this loop:
   `for (;;)  /* ... */`

Question 4 (Flowchart to Code). Write C++ code with branch and loop statements (possibly, including ”break” and ”continue”) to implement the flowchart shown in the picture.



Question 5 (Code to Flowchart).

For the code snippet below draw an equivalent flowchart. Does the "continue" statement jump to the bottom of the do-while loop (and retests the condition); or does it jump to the top of the do-while loop? If you pass "null" user to the method `isLast()` the program might crash.

```
userDao.init();
do {
  user = userDao.getNext();
  if (user == null) { continue; }
}
while (!user.isLast())
```

Please draw the flowchart nodes accurately: Use only 5 kinds of nodes:

(1) Start node (oval: one outgoing arrow).

(2) Stop node (oval: one incoming arrow).

(3) Conditional statement (diamond: one incoming and two outgoing arrows). Also mark the branch taken on `true`.

(4) Regular statement (rectangle: one incoming and one outgoing arrow).

(5) Merging two branches (black dot: two incoming arrows, one outgoing arrow).

Solutions

Question 1. Answer:

Variable	Hex value
a	FFFFFFFFFFFFFFFA
b	000000E7
c	09

To find `a`, note that -1 has hex representation `FFFFFFFFFFFFFFFF` (`long long` is 8 bytes long). To find -6, we subtract number 5. Moreover, `F` $-$ 5 = `A` (in hex), since 15 – 5 – 10 (in decimal).

To find `b`, we transform from octal to hexadecimal through binary. Octal number 0347_8 is the same as $000.011.100.111_2$.

Regroup the same bits (by four):

$0000.1110.0111_2 = 0E7_{16}$. Type `int` is 4 bytes long.

To find `c` look up the "TAB" character in the ASCII table `http://www.asciitable.com/`. It is the 9th byte that is written as `09`, since `char` is 1 byte long.

Question 2. Answer:

Variable	Hex value
a	0000002F
b	FFFFFFF3
c	63
Line 5	35
Line 7	-1
Line 9	12

```
a  = 0000.0000.0000.0000.0000.0000.0010.1111
b  = 1111.1111.1111.1111.1111.1111.1111.0011
============================================
L5 = 0000.0000.0000.0000.0000.0000.0010.0011
L7 = 1111.1111.1111.1111.1111.1111.1111.1111
L9 = 0000.0000.0000.0000.0000.0000.0000.1100
```

In order to find the output for lines 5,7,9, we can rewrite the hexadecimal content of `a`, `b` as binary.

Line5 contains bit "1" in all those positions, where both `a` AND `b` contains bit "1".

Line7 contains bit "1" where either `a` OR `b` contains bit "1".

Line9 contains bit "1" where `b` contains bit "0" and vice versa.

Question 3. Answer:

1. It is possible to change the loop variable inside `for` (this is certainly not recommended, because it violates usual intuition about the ”for” loops).

2. The value `i` is usable after the body of loop, if that variable is not declared inside the ”for” loop statement itself, but it is declared in a higher scope.

3. It is possible to omit any of the three parts in the ”for” loop. If you skip all three, it is an infinite loop.

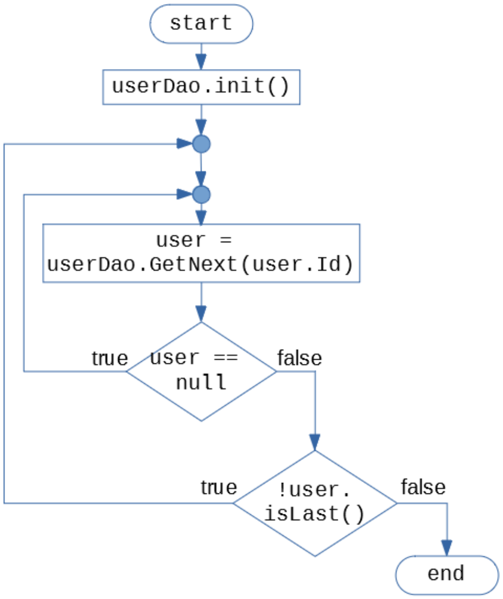
Question 4. Answer:

```
int i, j;
for(i = 1; i <= 10; i = i+1) {
  cout << i;
  for (j = i+1; j <= 10; j = j+1) {
    cout << j;
  }
}
```

The flowchart represents two nested ”for” loops. Flowchart does not enforce ”for” loops; it is possible to rewrite them as ”while” loops as well.

```
int i, j;
i = 1;
while (i <= 10) {
  cout << i;
  j = i+1;
  while (j <= 10) {
    cout << j;
    j = j+1;
  }
  i = i+1;
}
```

Question 5. Answer:


start
userDao.init()
user =
userDao.GetNext(user.Id)
true
user ==
null
false
true
!user.
isLast()
false
end