Tutorial Sheet (August 31, 2018)

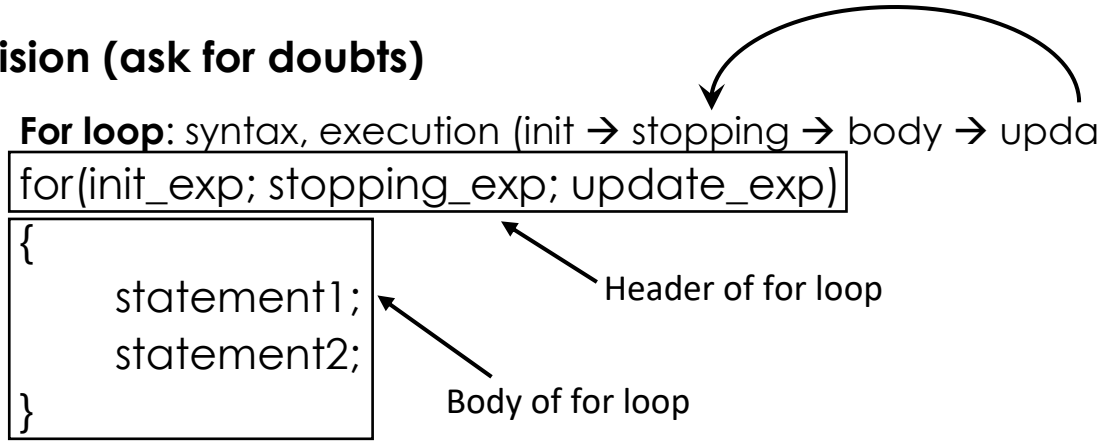
ESC101 – Fundamentals of Computing

Announcements

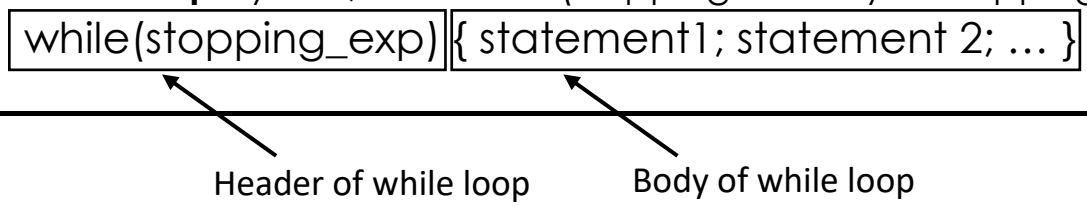
1. **Holiday** on September 03, 2018 (Monday) – no lecture, no labs
2. **Extra lecture**, September 08, 2018 (Saturday), L20, 12PM
3. **Extra lab for B1, B2, B3,** September 08, 2018, CC-01,CC-02, 2PM
4. **Mid-sem lab exam**, September 09, 2018 (details soon).
5. Students are advised to write legible, properly indented, commented code with meaningful variable names in the labs. TAs will now ask students to tidy up their code, if untidy with sloppy indentation, before helping them with their problems.

Revision (ask for doubts)

1. **For loop**: syntax, execution (init → stopping → body → update)



2. **Init** can be any arithmetic/relational/ternary/empty expression. If empty, put init before writing for loop.
3. **Update** can be arithmetic/relational/ternary/empty expression. If empty, put update inside body.
4. **Stopping** expression is a relational expression – can be empty.
5. **While loop**: syntax, execution (stopping → body → stopping …)



Tips for using loops: Ascendo

Take an input integer n and print the pattern 1 2 3 4 … n

1. Identify the same or similar sub-task that needs to be done repeatedly. E.g. here repeated task can be: *Print an integer i.* This task is repeated for various values of i from 1 to n.
2. The subtask should be expressible as a simple piece of code. E.g. to print an integer i, we simply write printf("%d", i);
3. **Variable of the loop**: i since it keeps changing
4. **Loop invariant**: in i-th iteration, integer i is printed
5. **Final code**: for(i = 1; i <= n; i++) printf("%d ", i); (note space)
6. **Challenge**: avoid printing space after last number

   Solution:

```
for(i = 1; i < n; i++)
        printf("%d ", i);
if(n>=1) printf("%d", n);
```

Not executed inside loop since no braces – executed only once no matter what n

if(n>=1) needed for n<=0 case

```
1
1 2
1 2 3
1 2 3 4
…
1 2 3 4 … n
```

Tips for using loops: Crescendo

Take an input integer n and print the pattern

1. Here, repeated task can be: *Print the sequence 1 2 3 … i* This task is repeated for various values of i from 1 to n. Note that this is simply the task Ascendo.
2. The subtask should be expressible as a simple piece of code. We have already seen that to print numbers from 1 to i with spaces between numbers but no space at the end
   for(k = 1; k < i; k++) printf("%d ", k); printf("%d", i);
3. **Variable of the loop**: k since it keeps changing
4. **Loop invariant**: in k-th iteration, 1 2 3 … k is printed (new line?)
5. **Final code**: (braces needed for outer loop – multiple statements)

```
for(i = 1; i <= n; i++){
        for(k = 1; k < i; k++) printf("%d ", k);
        printf("%d", i);
        printf("\n");
}
```

Sample Questions to discuss

Print multiplication tables of 2 to 9 in parallel (tab separator)

Quiz: define the repeated subtask, variable of the loop, and invariant

```
2 x 1 = 2   3 x 1 = 3   …   9 x 1 = 5
2 x 2 = 4   3 x 2 = 6   …   9 x 2 = 18
2 x 3 = 6   3 x 3 = 9   …   9 x 3 = 27
…
```

```
for(i = 1; i <= 10; i++){
    for(j = 2; j <= 9; j++)
        printf("%d x %d = %d\t", j, i, j * i);
    printf("\n");
}
```

Quiz: how do we get rid of extra tab at the end of every line?

Keep reading positive numbers till encounter -1 and print YES if the numbers seen so far form a non-decreasing sequence else print NO

Tip: if number of iterations not known, while loop elegant choice No explicit loop variable/counter variable as unknown no. of iterations

```
int num, prev = -1 // Used to store current and previous number
int isNonDec = 1; // Initially be optimistic
scanf("%d", &num);
while(num != -1){
    if(num < prev) // We initially set prev = -1 to be safe
        isNonDec = 0; // The numbers broke my trust ☹
    prev = num; // The current num is prev for next number
    scanf("%d", &num); // Read a new number
}
if(isNonDec) printf("YES"); else printf("NO");
```

Some Pitfalls and recognizing compiler error messages

1. In a for loop update executed after body executed. Be careful of this if inserting update statement inside body.
2. **Loop errors**: errors in header, forgotten braces, infinite loop (TLE)
