C: functions

COSC 208, Introduction to Computer Systems, 2021-09-03

Announcements

- Complete readings and pre-class questions before each class period
- Lab 1 due Friday, 11 p.m.

Outline

- Warm-up
- Defining functions
- Program stack

Warm-up

This warm-up focuses on the biological process of cell division: each cell splitting into two each round.

- Q1: *Write a function called `cells` that takes the number of rounds of cell division that occur and computes the total number of cells that will exist after the specified number of rounds (assuming you started with a single cell).*

```
```

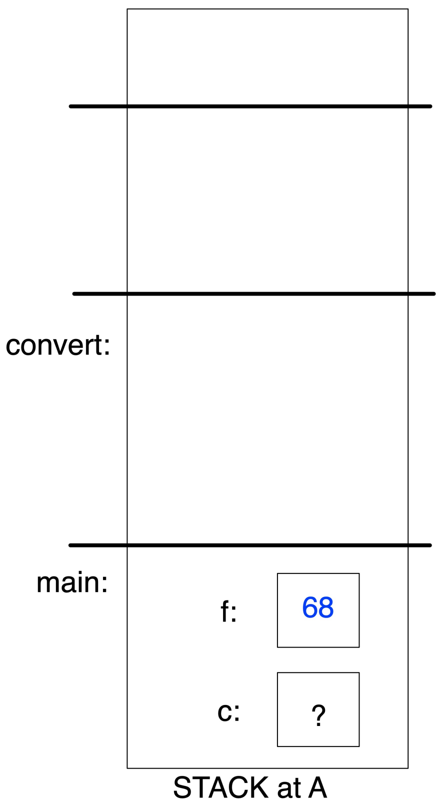
- Q2: *Write a function called `rounds` that takes the number of cells that should exist and computes the number of rounds of cell division that must occur to have ta least that many cells (assuming you started with a single cell).*

```
```

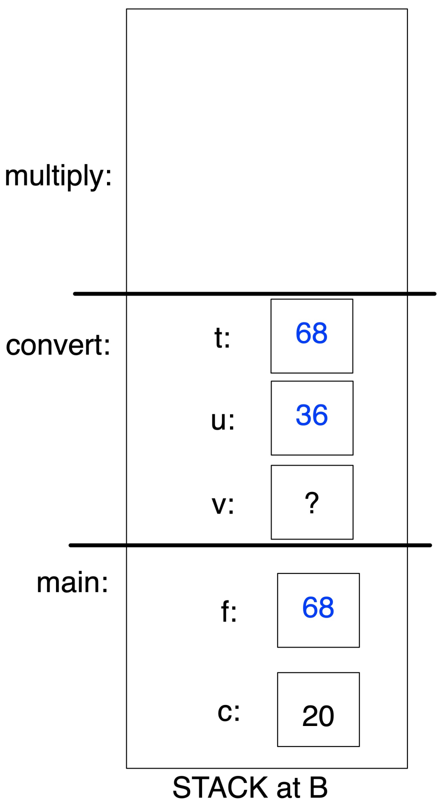
Program stack

```c
#include <stdio.h>
int multiply(int z) {
    return z * 5 / 9;      // stack at A
}
int subtract(int x, int y) {
    return x - y;
}
int convert(int t) {
    int u = subtract(t, 32);
    int v = multiply(u);
    return v;              // stack at B
}
int main() {
    int f = 68;
    int c = convert(68);
    printf("%dF is %dC\n", f, c);
}
```

- Example: Completing the drawings of stack frames at comment A on the left and at comment B on the right.



STACK at A



STACK at B

- Q3: *Draw the contents of the stack immediately before the program prints "1 x 2"*

```
int squared(int base) {
    return base * base;
}

int dbl(int num) {
    printf("%d x 2\n", num);
    return num * 2;
}

int two(int exponent) {
    int result = 1;
    for (int i = 0; i < exponent; i++) {
        result = dbl(result);
    }
    return result;
}

int main() {
    int n = 3;
    int s = squared(3);
    printf("%d^2 is %d\n", n, s);
    int t = two(3);
    printf("2^%d is %d\n", n, t);
}
```

- Q4: *What is the output of this program?*

```
int copy(int a, int b) {
    a = b;
    return b;
}
int main() {
    int x = 3;
    int y = 7;
    int z = copy(x, y);
    printf("%d %d %d\n", x, y, z);
}
```

Q5: *Draw the contents of the stack immediately before the program prints "n=2"*

```c
int recurse(int n) {
    printf("n=%d\n", n);
    if (n == 1) {
        return 0;
    }
    else {
        return 1 + recurse(n/2);
    }
}
int main() {
    int x = 16;
    int r = recurse(x);
    printf("result=%d\n", r);
}
```

Q6: *If `main` initialized `x` to `64` (instead of `16`), how many stack frames would exist immediately before the program printed "n=2"?*

Extra practice

Q7: *Write a function called `print_x` that takes an integer n that prints the letter `X` n lines high and n characters wide. You can assume n is an odd number. For example, if n = 9, the program's output would be:*

```
\       /
 \     /
  \   /
   \ /
    X
   / \
  /   \
 /     \
/       \
```

Worksheet created by Professor Aaron Gember-Jacobson