Computer Science 50: Introduction to Computer Science I
Harvard College
Fall 2008

Quiz 1

Solutions

Answers other than the below may be possible.

O(hai).

0.

	O	Ω
inserting an integer into a sorted doubly linked list	*n*	1
inserting an integer into a sorted singly linked list	*n*	1
sorting an unsorted array of integers with heapsort	*n* log *n*	*n* log *n*

Bits.

1.

```
  10111111
| 00000001
----------
  10111111
```

2.

```
  10111111
^ 00000001
----------
  10111110
```

3. `50`

4. `true`

Trees.

5. The one on the right.

6.
```
typedef struct node
{
    int n;
    struct node *left;
    struct node *right;
}
node;
```

7.
```
bool
find(node *tree, int n)
{
    if (tree == NULL)
        return false;
    else if (tree->n == n)
        return true;
    else if (find(tree->left, n))
        return true;
    else if (find(tree->right, n))
        return true;
    else
        return false;
}
```

8.
```
bool
insert(node *tree, int n)
{
    if (tree == NULL)
        return false;

    if (n < tree->n)
    {
        if (tree->left == NULL)
        {
            tree->left = malloc(sizeof(node));
            if (tree->left == NULL)
                return false;
            tree->left->n = n;
            tree->left->left = NULL;
            tree->left->right = NULL;
        }
        else
            return insert(tree->left, n);
    }
    else if (tree->n < n)
    {
        if (tree->right == NULL)
        {
            tree->right = malloc(sizeof(node));
            if (tree->right == NULL)
                return false;
            tree->right->n = n;
            tree->right->left = NULL;
            tree->right->right = NULL;
        }
        else
            return insert(tree->right, n);
    }
    else
        return false;

    return true;
}
```

Rapid Fire.

9. Maybe.
10. It stores multi-byte values' little ends (*i.e.*, least-significant bytes) at lower memory addresses.
11. When user-supplied input is not escaped before use in a SQL query.
12. Uniformly distributing some (possibly non-uniform) domain over a range.
13. For pattern matching, including, perhaps, extraction of substrings.
14. Maybe.

Alert!

15. Rather than assign to the button's `onlick` property the address of a function, this line of code instead assigns to that property the return value (if any) of `alert`, which means that `alert` is actually called (prematurely) when this line executes rather than being registered for execution upon actual clicks.

16. Because these lines of code do not execute atomically (*i.e.*, together without interruption or not at all), it's possible for money to be dispensed without the database being updated, as might happen if, say, the ATM loses power right after the call to `dispense`.[1] Alternatively, if Lord Dark Helmet has access to two ATMs (say, side-by-side), he could attempt to withdraw money from both simultaneously in hopes that `SELECT` will be executed on both before either `UPDATE`, thereby exploiting a race condition.

Time for Change.

17.
```
function change($amount)
{
    $coins = 0;
    $cents = round($amount * 100);
    while ($cents > 0)
    {
        if ($cents >= 25)
            $cents -= 25;
        else if ($cents >= 10)
            $cents -= 10;
        else if ($cents >= 5)
            $cents -= 5;
        else
            $cents -= 1;
        $coins++;
    }
    return $coins;
}
```

[1] Or if Lord Dark Helmet himself pulls the cord!

What are you looking 404?

18.
```
<form action="index.php" method="get">
  <input name="from" type="text" />
  <input type="submit" />
</form>
```

O hai again, C.

19. **During pre-processing...** directives like `#include` are processed, which, in this case, means that

```
#include <stdio.h>
```

is effectively replaced with the contents of `stdio.h`, thereby providing a prototype for `printf`.

During compiling... the pre-processed C code is translated into assembly instructions, possibly with optimizations applied.

During assembling... the assembly instructions are translated into machine code and stored in an object (`.o`) file.

During linking... the object code for `main` is combined with (*i.e.*, linked against) that for `printf`, the result of which is an executable file.

Design Decisions.

Below are more arguments than were expected. Yet more are possible.

20. C, a compiled language, should probably be chosen over PHP, an interpreted language, when performance is particularly important, as compiled programs tend to execute more quickly than interpreted ones.

21. Arrays should probably be chosen over linked list when random access to elements is more important (for, say, performance reasons) than the data structure's ability to grow dynamically. Or when you know in advance the size of your data.

22. CSV should probably be chosen over XML when data lends itself to representation in rows and (fixed numbers of) columns or when bandwidth or space is limited, thereby warranting CSV's more compact representation.

XML should be chosen over CSV when extensibility (the ability of data to grow beyond fixed numbers of columns) is important. Or when human-readability is important, as XML embeds alongside the data a good deal of metadata.

23. Hexadecimal should be chosen over decimal when you want each digit to represent exactly four bits (*e.g.*, to define bitmasks).

    Decimal should probably be chosen over hexadecimal when you want (non-technical) humans to understand numbers.

24. POST should be chosen over GET when large amounts of data are to be submitted from client to server. Or when sensitive data (*e.g.*, passwords) need to be submitted.

    GET should be chosen over POST when you want to embed state in a URL (so as to facilitate bookmarking or emailing pages' addresses).

25. Horizontal scaling should be chosen over vertical scaling when an architecture needs to scale, perhaps long-term, beyond the capabilities of individual (if top-of-the-line) systems. Or when buying multiple, lower-end systems is more cost-effective than buying one, high-end system. Or when an architecture lends itself to distribution across multiple systems.

DOM, DOM DOM DOM, DOM.

26.
```
<rss version="0.92">
  <channel>
    <title>CS50 News</title>
    <description>This just in...</description>
    <link>http://cs50.net/</link>
    <item>
      <description>Teh End</description>
    </item>
  </channel>
</rss>
```