Quiz 5 Version A

Due No due date	**Points** 8	**Questions** 4	**Available** after Nov 9 at 3:30pm
Time Limit None	**Allowed Attempts** Unlimited		

Instructions

Quiz for Lecture 5: Recursion

7 points required to pass

Take the Quiz Again

Attempt History

	Attempt	Time	Score
LATEST	Attempt 1	64 minutes	5 out of 8

Score for this attempt: **5** out of 8
Submitted Nov 9 at 4:44pm
This attempt took 64 minutes.

Question 1

1 / 1 pts

Recursion should be as easy for you as

- Building a Mars rocket from spare parts
- Performing heart surgery on yourself
- [✓ Iteration] Correct!
- Finding a sunken Spanish treasure ship in the Carribean

Question 2

2 / 2 pts

The Fibonacci sequence can be defined as:

Fib(0) == 1

Fib(1) == 1

Fib(n) == Fib(n-2) + Fib(n-1) for n >= 2

Write a recursive implementation of the Fibonacci sequence function, Fib:

```
unsigned Fib(unsigned n)
{
   // your code goes here
}
```

Your Answer:

```
if ( n == 0 ) {
return 1;
}
else if ( n == 1 ) {
return 1;
}
else {
return Fib(n-1) + Fib(n-2);
}
```

if (n == 0) return 1;

if (n == 1) return 1;

return Fib(n - 2) + Fib(n - 1);

Question 3

2 / 2 pts

Write a recursive implementation of the function **put_str_rev**, so that this statement in **main**:

put_str_rev("hello");

would display

olleh

to the screen.

Your Answer:

void put_str_rev(string inStr) {

if (inStr.size() == 0)

cout << "" << endl;

}

else {

cout << inStr[inStr.size() - 1];

auto indx_last = inStr.size() - 1;

return put_str_rev(substring(inStr[0], indx_last))

}

```
void put_str_rev(const char *s)
{
    if (*s) {
        put_str_rev(s + 1);
        cout << *s;
    }
}
```

very inefficient, to make all these copies of substrings

Question 4

0 / 3 pts

Given this structure definition:

struct cl_node { char data; cl_node *next; };

Define a recursive function **mk_clist_from_Cstring_rec** that makes a singly linked list of **char**s from a C-string, and returns the address of the first **cl_node** in the list.

Your Answer:

cl_node* mk_clist_from_Cstring_rec (const char *str) {

```
cl_node* head{nullptr};

// base case

if ( str == nullptr ) {

return head; // but how to return the head of the linked list?

} // recursive call

head = new cl_node{ str[0].data, nullptr };

auto i = 0;

while ( str[i] != nullptr ) {

head->next = new cl_node{ str[i].data, nullptr };

return mk_clist_from_Cstring_rec( str);

}
```

```
cl_node *mk_clist_from_Cstring_rec(const char *s)

{

   if (*s == '\0')

      return nullptr;

   return new cl_node{ *s, mk_clist_from_Cstring_rec(s + 1) };

 }
```

str is a pointer-to char, so str[i].data is not meaningful some other issues

Quiz Score: **5** out of 8