1 Quick Review

```
1. Which of the following is the proper declaration of a pointer?
A. int x;
B. int &x;
C. ptr x;
D. int *x;

2. Which of the following gives the memory address of integer variable a?
A. *a;
B. a;
C. &a;
D. address(a);

3. Which of the following gives the memory address of a variable pointed to by pointer a?
A. a;
B. *a;
C. &a;
D. address(a);

4. Which of the following gives the value stored at the address pointed to by pointer a?
A. a;
B. val(a);
C. *a;
D. &a;
```

2 Poor Man's Prefix Sum

Write a function `void prefix_sum (int *a, int *b, int *c)` that takes in three integer pointers and while it returns nothing, the value at `a` will remain `a`; the value at `b` will be `a + b`, and the value at `c` will be `a + b + c`. Test your code by calling it from another function.

3 String Length

`strlen(s)` returns the length of a C string. As we discussed, a C string is terminated using a special character, the ASCII `'\0'`. Write a function `int my_str_len(char *s)` that computes the length of a C string, without calling `strlen`. To practice using pointers, do not use array notation (i.e., no `s[]`)—work with pointers directly.

4 More Pointers (Extra)

For each of the following functions, use the pointer notation **only**. Do **not** use the array index `[]` notation.

(i) Write a piece of code which prints the characters in a C string in a reverse order. Here's some starter code.

```c
char s[10] = "abcde";
char* cptr;

// WRITE YOUR CODE HERE
```

(ii) Write a function `int count_even(int*, int)`, which takes in an integer array and its size and returns the number of even numbers in the array.

(iii) Write a function `double* maximum(double* a, int size)` that returns a pointer to the maximum value of an array of `double`s. If the array is empty, return `NULL`.

(iv) Write a function `int contains(char*, char)` which returns true (i.e., nonzero) if the 1st parameter C string contains the 2nd parameter char. Otherwise, it returns false (i.e., 0).

(v) Write a function `void rev_str(char*)` which reverses the parameter C string. The function returns nothing. Some starter code:

```c
int main()
{
  char s[10] = "abcde";
  rev_str(s);  // call the function
  printf("%s\n", s); // s has been reversed
  return 0;
}

void rev_str(char* ptr)
{
  // WRITE YOUR CODE HERE
}
```