Mystery functions!
Which of the following is a problem with the code below?

```c
int* f(int x, double* y) {
    int z = 23;
    if(x < *y)
        z = (int) (z - *y);
    return &z;
}
```

A. Type error involving argument y
B. Type mismatch between signature and return value
C. Shouldn’t return a pointer to a local variable
D. Other syntax error
E. The code is fine (albeit not useful)
Which of the following is a problem with the code below?

```c
int* f(int x, double* y) {
    int z = 23;
    if(x < *y)
        z = (int) (z - *y);
    return &z;
}
```

A. Type error involving argument y
B. Type mismatch between signature and return value
C. Shouldn’t return a pointer to a local variable
D. Other syntax error
E. The code is fine (albeit not useful)
Pointers as arrays
Which of the following is **NOT** an advantage of C’s lack of bounds checking on arrays (relative to Java)?

A. Arrays in C support insertion (creating cells)
B. Arrays in C use less memory
C. Arrays in C can be accessed faster
D. More than one of the above
E. None of the above
Which of the following is **NOT** an advantage of C’s lack of bounds checking on arrays (relative to Java)?

A. Arrays in C support insertion (creating cells)
B. Arrays in C use less memory
C. Arrays in C can be accessed faster
D. More than one of the above
E. None of the above
Strings in C

• Just an array of chars with a 0 at the end (not ‘0’)
  – Type is a char*

• For “methods”, use functions from string.h

• Things to remember:
  – Allocate room for the \0
  – Scanf (with %s) takes a char*, not a char**