1 Errors in WLP4 and C

1.1 Errors in WLP4

For each WLP4 program below, point out the error in the program and state whether it is a syntax error (i.e. something the scanner or parser would catch) or a semantic error (something semantic analysis would catch).

```c
int wain(int x, int y) {
    int a = 100;
    int y = 0; // initialize y
    y = a*x;
    return y;
}
```

```c
int wain(int* a, int n) {
    // loop to get the last index
    while (idx < n) {
        idx = idx + 1;
    }
    return *(a + idx);
}
```

```c
int wain(int a, int b) {
    int *c = NULL;
    c = &a;
    int *d = NULL;
    d = &b;
    return (c - d);
}
```

```c
int wain(int x, int y) {
    int a = 'a';
    return a + x;
}
```
int sub(int a) {
    return a;
}

int sub(int a, int b) {
    return sub(a) + sub(b);
}

int wain(int x, int y) {
    int a = 0;
    a = sub(x, y);
    return *a;
}

int f(int a, int b) {
    return g(a) + g(b);
}

int g(int a) {
    return a + 32;
}

int wain(int x, int y) {
    return f(y, x);
}

1.2 Errors in C
For each C program below, point out the error in the program and state whether it is a syntax error (i.e. something the scanner or parser would catch) or a semantic error (something semantic analysis would catch).

float triple(float a) {
    return a * 3.0;
}

int main() {
    int* x, y;
    int a, b;

    a = triple(4.4);
    x = &a;
    y = &b;
    b = *x;
    return *y;
}
2 Symbol Tables and Type Checking

2.1 Symbol Tables

In the MIPS assembler you wrote for Assignments 3 and 4, you had to check for duplicate labels in one pass and check for missing labels in a second pass. Why are two passes not necessary in the WLP4 compiler?

2.2 Type Checking

For the WLP4 Code fragment given as follow:

```c
int foo(int x, int y)
{
    return x + 7 * y + 1;
}

int a = 0;
int b = 0;
int* c = NULL;
int* d = NULL;
```

Determine if each of the following statements are well typed:

1. *(d+(((c-&b)+d)-(c+(a*b))))=(c-d+*new int[d+b-c]);

2. if(*(c+a%b)<(&a-&b)){println(& *& *c-(&b));}else{delete[]*d+& a-c;}

3. foo(foo(a,b), foo(*c, foo(&a-&b,*(c-a))))