4

1

Report the compiler errors in ~/csds312/debugging/add.c after:

a

commenting `#include <stdio.h>`

```c
// #include <stdio.h>          //directive to include function
declarations and macro definitions
int main()
{
    int a = 5;
    int b = 10;
    int result = a + b;
    printf("result=%d\n",result);
    return 0;
}
```

Gives the following error

```
[tln32@class-login2 debugging]$ gcc -o add add.c
add.c: In function ‘main’:
add.c:7:5: warning: implicit declaration of function ‘printf’ [-
Wimplicit-function-declaration]
     printf("result=%d\n",result);
     ^~~~~
add.c:7:5: warning: incompatible implicit declaration of built-in
function ‘printf’
add.c:7:5: note: include ‘<stdio.h>’ or provide a declaration of
‘printf’
add.c:1:1:
+#include <stdio.h>
 // #include <stdio.h>          //directive to include function
declarations and macro definitions
add.c:7:5:
     printf("result=%d\n",result);
     ^~~~~
```

b

eliminating semicolon in one of the statements

```
#include <stdio.h>          //directive to include function declarations
and macro definitions
int main()
{
    int a = 5;
    int b = 10;
    int result = a + b;
    printf("result=%d\n",result);
    return 0
}
```

Gives the following error

```
[tln32@class-login2 debugging]$ gcc -o add add.c
add.c: In function ‘main’:
add.c:8:13: error: expected ‘;’ before ‘}’ token
     return 0
             ^
             ;
 }
 ~
```

2

```
[tln32@class-login2 debugging]$ cd /usr/local/doc/BOOTCAMP/bootcamp/
[tln32@class-login2 bootcamp]$ srun -A csds312 -p markov_cpu --pty
/bin/bash
[tln32@classct015 bootcamp]$ time ./prime
Number of Prime Numbers from 2 to 131072=12251

real    0m1.962s
user    0m1.955s
sys 0m0.001s
[tln32@classct015 bootcamp]$ module load Python/3.12.3-GCCcore-13.3.0
[tln32@classct015 bootcamp]$ time python primeNumbers.py
Number of Prime Numbers =  12251
Execution Time =  80.74629473686218

real    1m20.894s
user    1m20.569s
sys 0m0.024s
```

C shows better performance at this test, this is because C is compiled and Python is interpreted, meaning each loop, the code will need to be reunderstood again, while C's code turns into lower level abstractions before running.