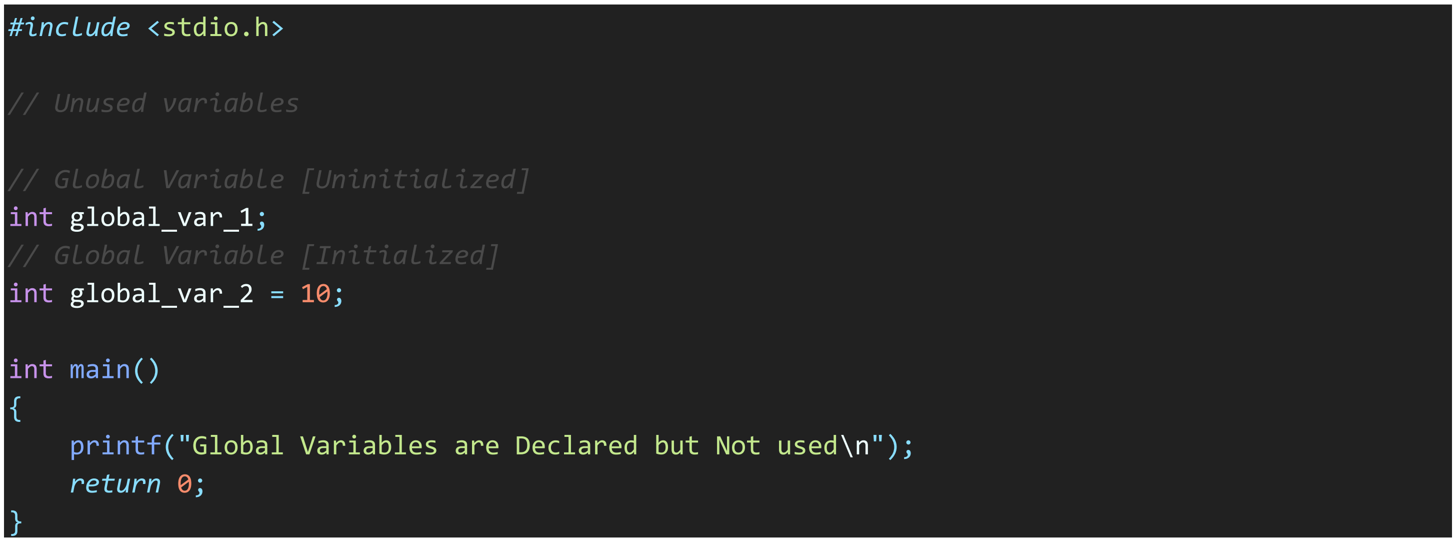
Software Engineering (CS401)

Lab Assignment 2

U19CS012

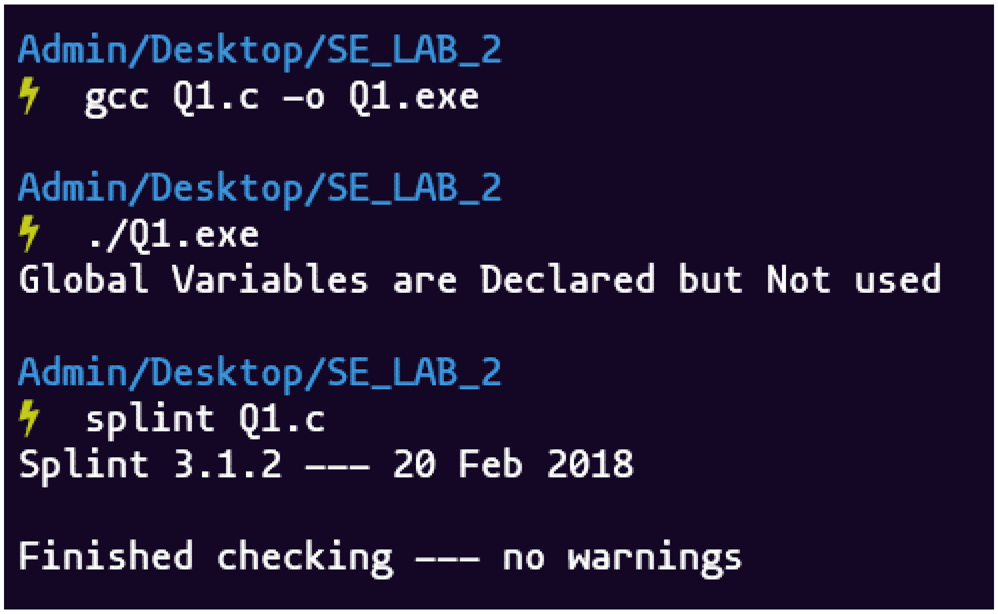
Q1.) Write a **C program** having some **global variables** that are **declared** but **not used** anywhere in the code. Run Splint for this C code and report the error generated.

Code



```c
#include <stdio.h>

// Unused variables

// Global Variable [Uninitialized]
int global_var_1;
// Global Variable [Initialized]
int global_var_2 = 10;

int main()
{
    printf("Global Variables are Declared but Not used\n");
    return 0;
}
```

Output

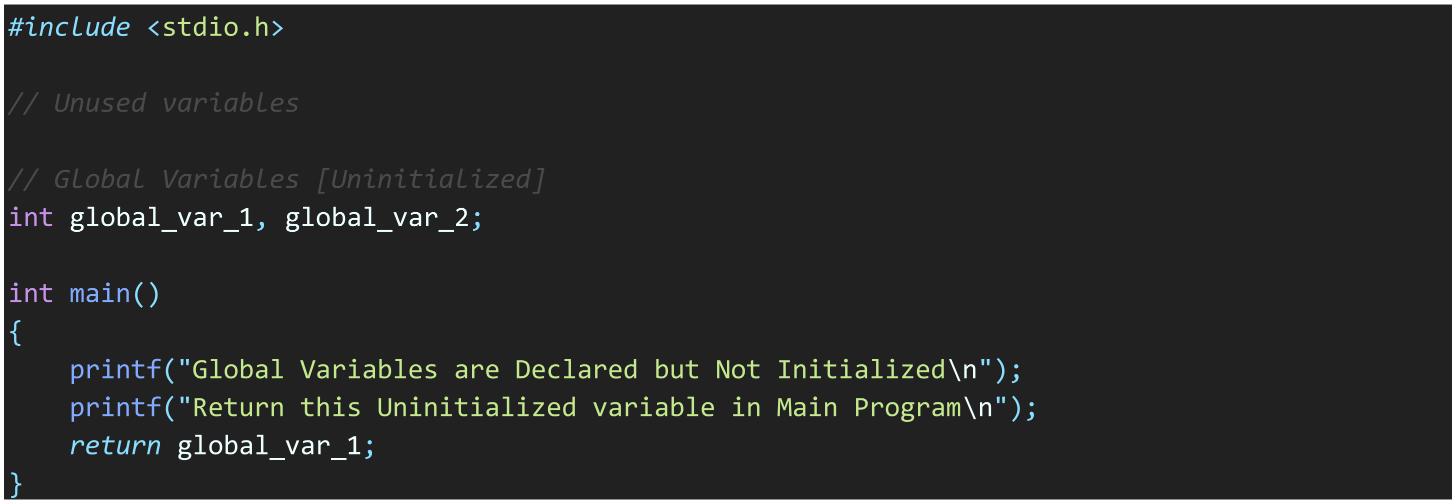


```
Admin/Desktop/SE_LAB_2
ϟ  gcc Q1.c -o Q1.exe

Admin/Desktop/SE_LAB_2
ϟ  ./Q1.exe
Global Variables are Declared but Not used

Admin/Desktop/SE_LAB_2
ϟ  splint Q1.c
Splint 3.1.2 --- 20 Feb 2018

Finished checking --- no warnings
```

No Error Generated by Splint Tool.

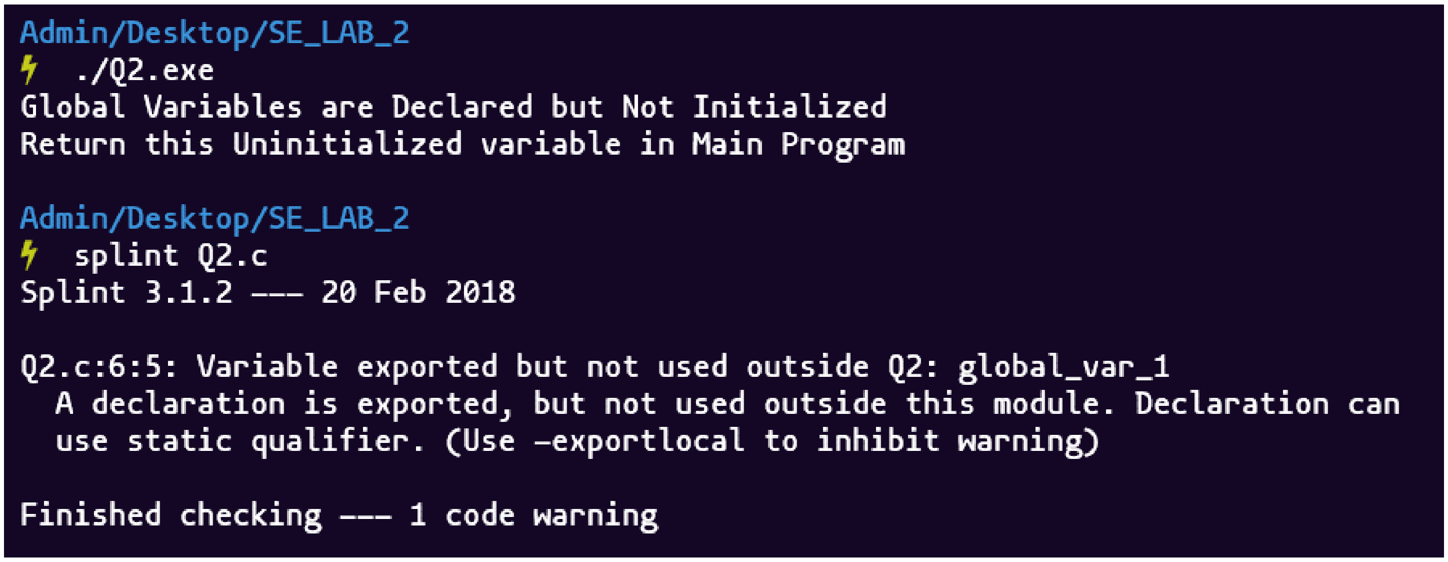
Q2.) Write a **C program** having some **global variables** that are **declared** but **not initialized**. Return this **uninitialized variable** in the main function. Run Splint for this C code and report the error generated.

Code



```c
#include <stdio.h>

// Unused variables

// Global Variables [Uninitialized]
int global_var_1, global_var_2;

int main()
{
    printf("Global Variables are Declared but Not Initialized\n");
    printf("Return this Uninitialized variable in Main Program\n");
    return global_var_1;
}
```

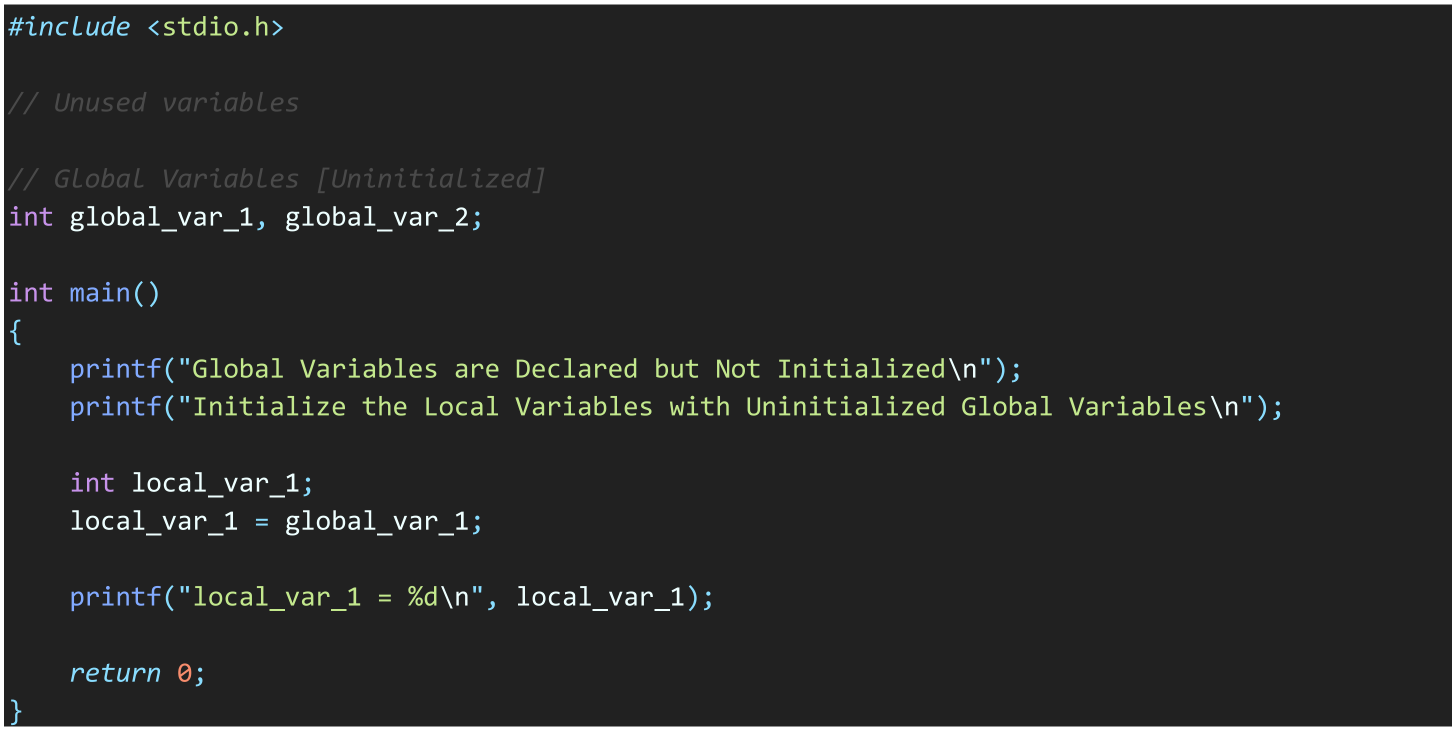
Output



```
Admin/Desktop/SE_LAB_2
ϟ  ./Q2.exe
Global Variables are Declared but Not Initialized
Return this Uninitialized variable in Main Program

Admin/Desktop/SE_LAB_2
ϟ  splint Q2.c
Splint 3.1.2 --- 20 Feb 2018

Q2.c:6:5: Variable exported but not used outside Q2: global_var_1
  A declaration is exported, but not used outside this module. Declaration can
  use static qualifier. (Use -exportlocal to inhibit warning)

Finished checking --- 1 code warning
```

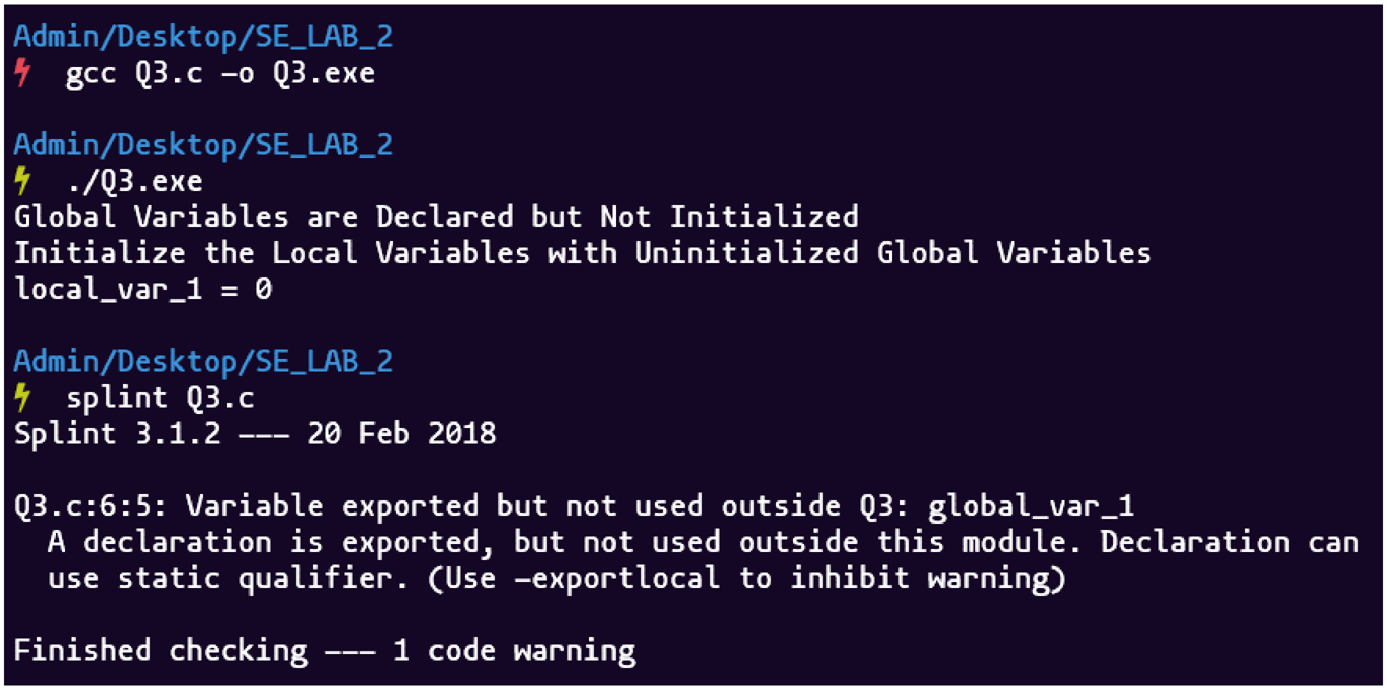
Q3.) Write a **C program** having some **global variables** that are **declared** but **not initialized**. Initialize some local variable using this **uninitialized global variable**. Run Splint for this C code and report the error generated. (For instance, assume global variable 'a' is declared as 'int' in the code. In the main function you can perform some operation such as 'int b =a'. This code should generate some error as the variable 'a' is not initialized in the code.)

Code



```c
#include <stdio.h>

// Unused variables

// Global Variables [Uninitialized]
int global_var_1, global_var_2;

int main()
{
    printf("Global Variables are Declared but Not Initialized\n");
    printf("Initialize the Local Variables with Uninitialized Global Variables\n");

    int local_var_1;
    local_var_1 = global_var_1;

    printf("local_var_1 = %d\n", local_var_1);

    return 0;
}
```

Output

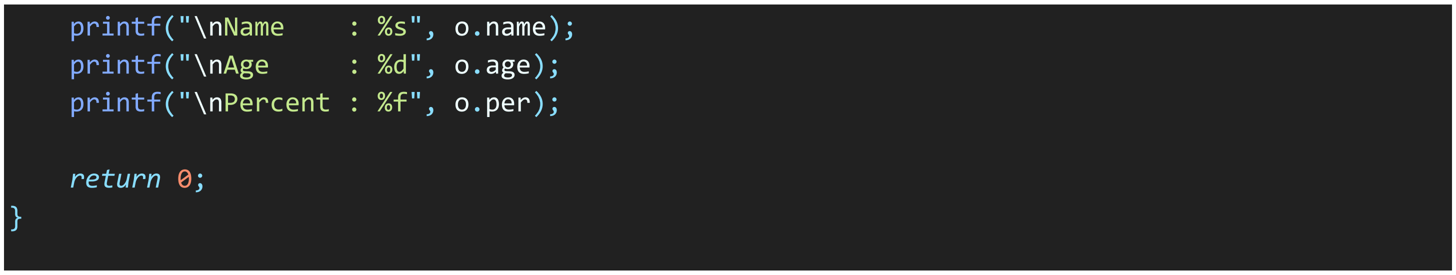


```
Admin/Desktop/SE_LAB_2
ϟ  gcc Q3.c -o Q3.exe

Admin/Desktop/SE_LAB_2
ϟ  ./Q3.exe
Global Variables are Declared but Not Initialized
Initialize the Local Variables with Uninitialized Global Variables
local_var_1 = 0

Admin/Desktop/SE_LAB_2
ϟ  splint Q3.c
Splint 3.1.2 --- 20 Feb 2018

Q3.c:6:5: Variable exported but not used outside Q3: global_var_1
  A declaration is exported, but not used outside this module. Declaration can
  use static qualifier. (Use -exportlocal to inhibit warning)

Finished checking --- 1 code warning
```

Q.4) Write a **C program** having **structure** as **global variable**. This structure can have more than two fields. Except one field, you can **initialize values** to all fields in the structure. Run Splint for this C code and report the error generated. (This code should generate error as you have one uninitialized field in structure)

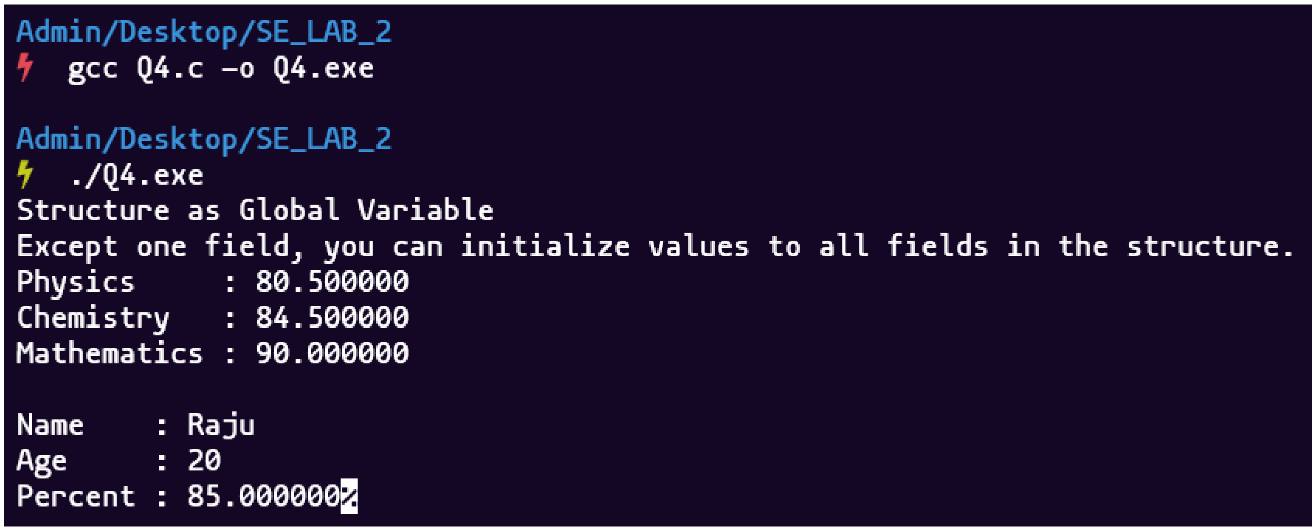
Code

```c
#include <stdio.h>

// Global Structure
struct student
{
    char *name;
    int age;
    float per;
};

float get_percent()
{
    // Local Structure
    struct mark
    {
        float m1, m2, m3;
    } s;

    s.m1 = 80.5;
    s.m2 = 84.5;
    s.m3 = 90;
    printf("Physics     : %f\n", s.m1);
    printf("Chemistry   : %f\n", s.m2);
    printf("Mathematics : %f\n", s.m3);

    float percent = (s.m1 + s.m2 + s.m3) / 3;
    return percent;
}

int main()
{
    printf("Structure as Global Variable\n");
    printf("Except one field, you can initialize values to all fields in the structure.\n");

    struct student o =
        {
            .name = "Raju",
            .age = 20,
        };

    o.per = get_percent();
```

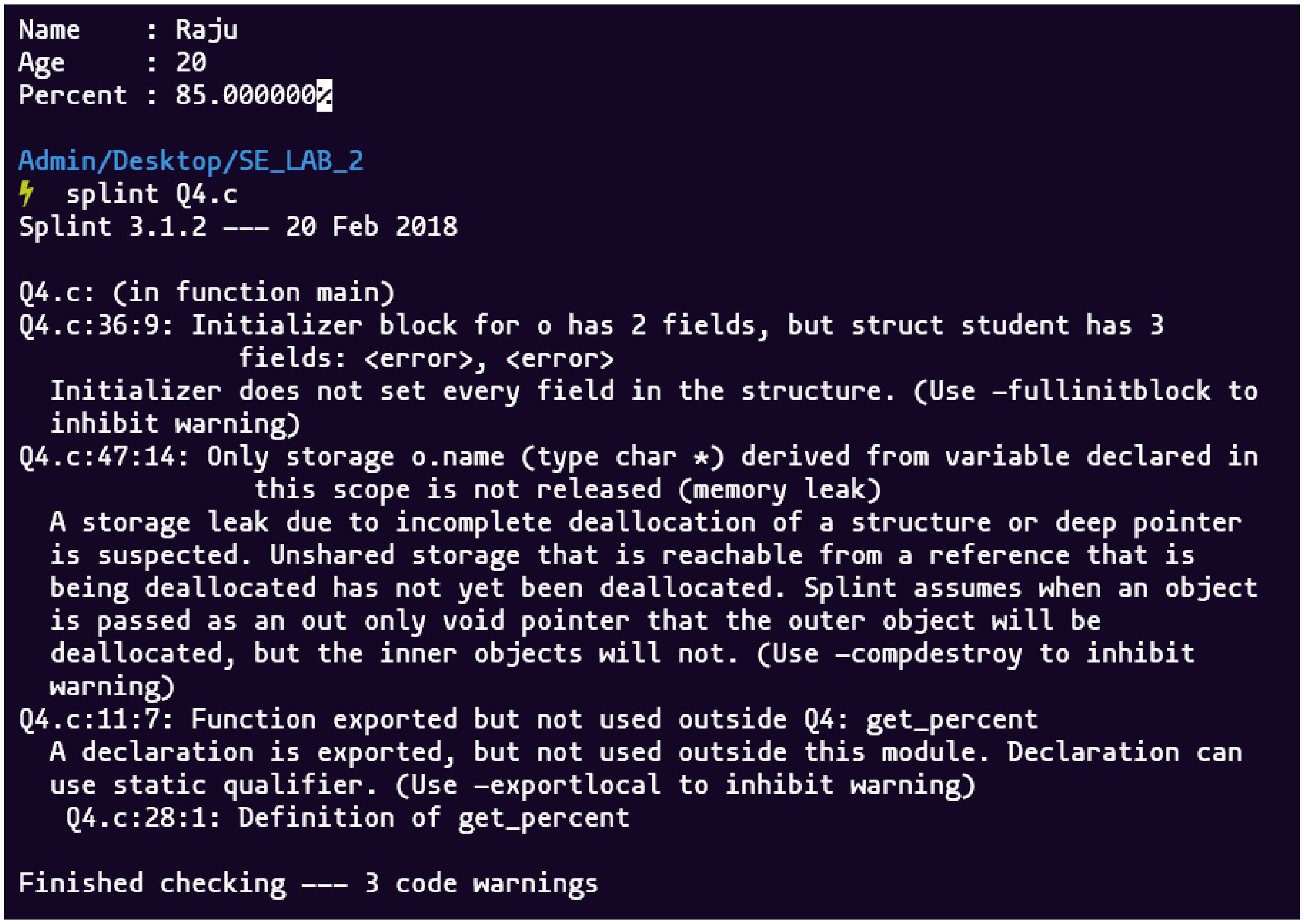


```c
    printf("\nName    : %s", o.name);
    printf("\nAge     : %d", o.age);
    printf("\nPercent : %f", o.per);

    return 0;
}
```

Output



```
Admin/Desktop/SE_LAB_2
ϟ  gcc Q4.c -o Q4.exe

Admin/Desktop/SE_LAB_2
ϟ  ./Q4.exe
Structure as Global Variable
Except one field, you can initialize values to all fields in the structure.
Physics     : 80.500000
Chemistry   : 84.500000
Mathematics : 90.000000

Name    : Raju
Age     : 20
Percent : 85.000000%
```



```
Name    : Raju
Age     : 20
Percent : 85.000000%

Admin/Desktop/SE_LAB_2
ϟ  splint Q4.c
Splint 3.1.2 --- 20 Feb 2018

Q4.c: (in function main)
Q4.c:36:9: Initializer block for o has 2 fields, but struct student has 3
              fields: <error>, <error>
  Initializer does not set every field in the structure. (Use -fullinitblock to
  inhibit warning)
Q4.c:47:14: Only storage o.name (type char *) derived from variable declared in
               this scope is not released (memory leak)
  A storage leak due to incomplete deallocation of a structure or deep pointer
  is suspected. Unshared storage that is reachable from a reference that is
  being deallocated has not yet been deallocated. Splint assumes when an object
  is passed as an out only void pointer that the outer object will be
  deallocated, but the inner objects will not. (Use -compdestroy to inhibit
  warning)
Q4.c:11:7: Function exported but not used outside Q4: get_percent
  A declaration is exported, but not used outside this module. Declaration can
  use static qualifier. (Use -exportlocal to inhibit warning)
   Q4.c:28:1: Definition of get_percent

Finished checking --- 3 code warnings
```