Ryan Reynolds
2693018
CIS 340

Assignment 4

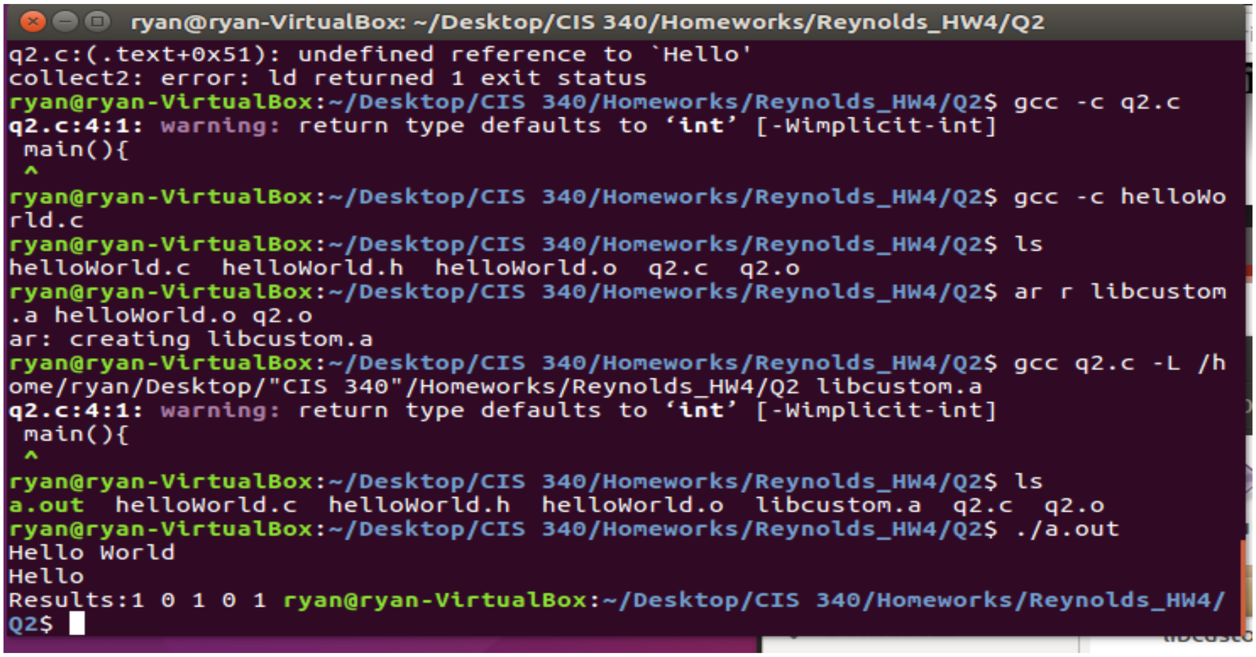
Question 1:

label	address	value
I (parent)	400-403	~~Child PID~~ 0
J (parent)	404-407	~~0 1 2 3~~ 4
K (parent)	408-411	~~0 0 1 3~~ 6
I (child)	500-503	~~0 3 4 5~~ 6
J (child)	504-507	~~4 1 -3~~ -8
K (child)	508-511	6

The output would either be
0 4 6
6 -8 6
or
6 -8 6
0 4 6

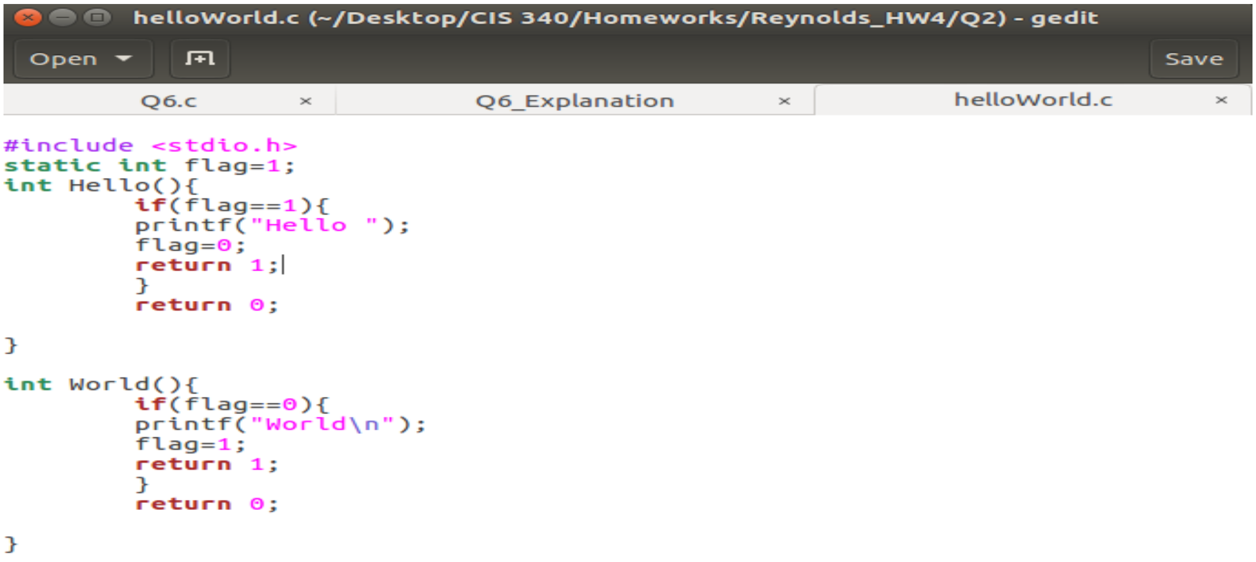
The program uses printf which buffers the variables. The processes execute concurrently and the output will be determined by the CPU scheduler i.e. the output depends on which process reaches the printf statement first which could be either for this case.

Question 2:

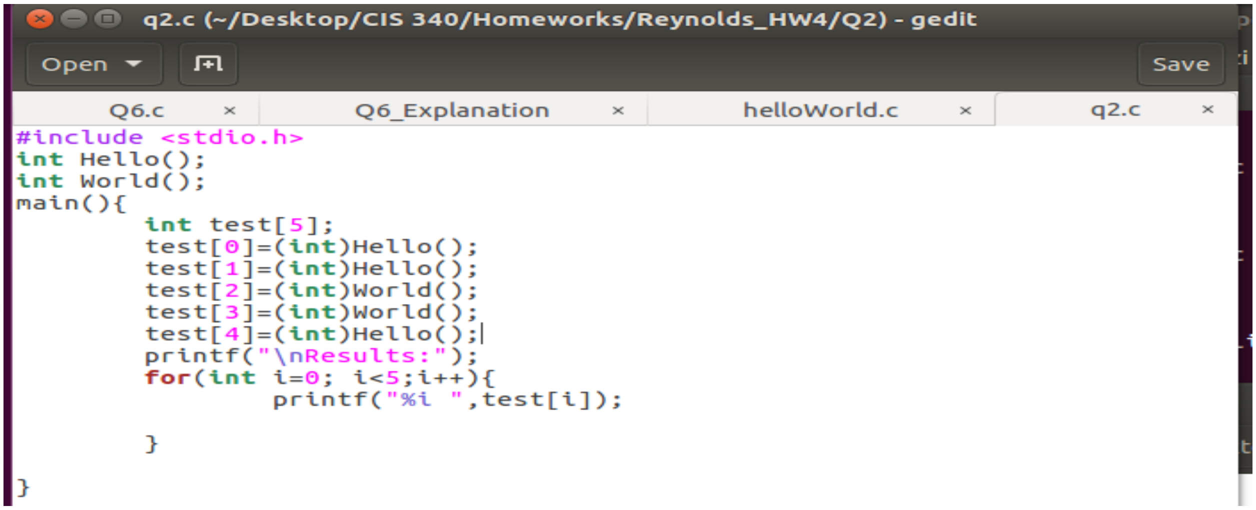


ryan@ryan-VirtualBox: ~/Desktop/CIS 340/Homeworks/Reynolds_HW4/Q2

```
q2.c:(.text+0x51): undefined reference to `Hello'
collect2: error: ld returned 1 exit status
ryan@ryan-VirtualBox:~/Desktop/CIS 340/Homeworks/Reynolds_HW4/Q2$ gcc -c q2.c
q2.c:4:1: warning: return type defaults to ‘int’ [-Wimplicit-int]
 main(){
 ^
ryan@ryan-VirtualBox:~/Desktop/CIS 340/Homeworks/Reynolds_HW4/Q2$ gcc -c helloWo
rld.c
ryan@ryan-VirtualBox:~/Desktop/CIS 340/Homeworks/Reynolds_HW4/Q2$ ls
helloWorld.c  helloWorld.h  helloWorld.o  q2.c  q2.o
ryan@ryan-VirtualBox:~/Desktop/CIS 340/Homeworks/Reynolds_HW4/Q2$ ar r libcustom
.a helloWorld.o q2.o
ar: creating libcustom.a
ryan@ryan-VirtualBox:~/Desktop/CIS 340/Homeworks/Reynolds_HW4/Q2$ gcc q2.c -L /h
ome/ryan/Desktop/"CIS 340"/Homeworks/Reynolds_HW4/Q2 libcustom.a
q2.c:4:1: warning: return type defaults to ‘int’ [-Wimplicit-int]
 main(){
 ^
ryan@ryan-VirtualBox:~/Desktop/CIS 340/Homeworks/Reynolds_HW4/Q2$ ls
a.out  helloWorld.c  helloWorld.h  helloWorld.o  libcustom.a  q2.c  q2.o
ryan@ryan-VirtualBox:~/Desktop/CIS 340/Homeworks/Reynolds_HW4/Q2$ ./a.out
Hello World
Hello
Results:1 0 1 0 1 ryan@ryan-VirtualBox:~/Desktop/CIS 340/Homeworks/Reynolds_HW4/
Q2$
```

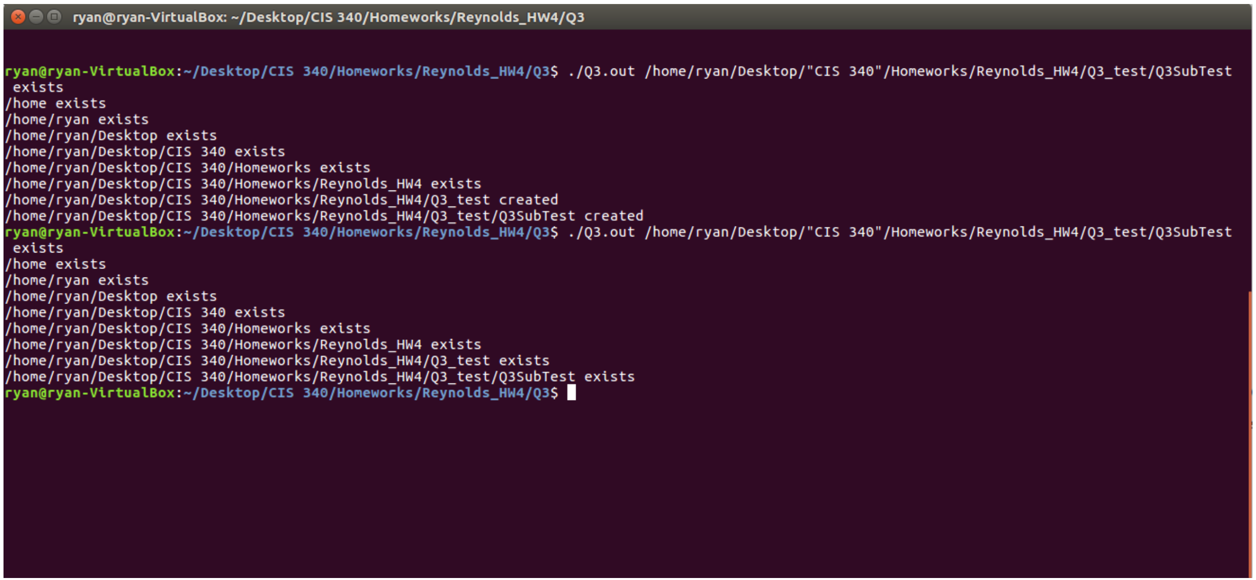

helloWorld.c (~/Desktop/CIS 340/Homeworks/Reynolds_HW4/Q2) - gedit
Open
Save
Q6.c
Q6_Explanation
helloWorld.c

```c
#include <stdio.h>
static int flag=1;
int Hello(){
        if(flag==1){
        printf("Hello ");
        flag=0;
        return 1;
        }
        return 0;

}

int World(){
        if(flag==0){
        printf("World\n");
        flag=1;
        return 1;
        }
        return 0;

}
```

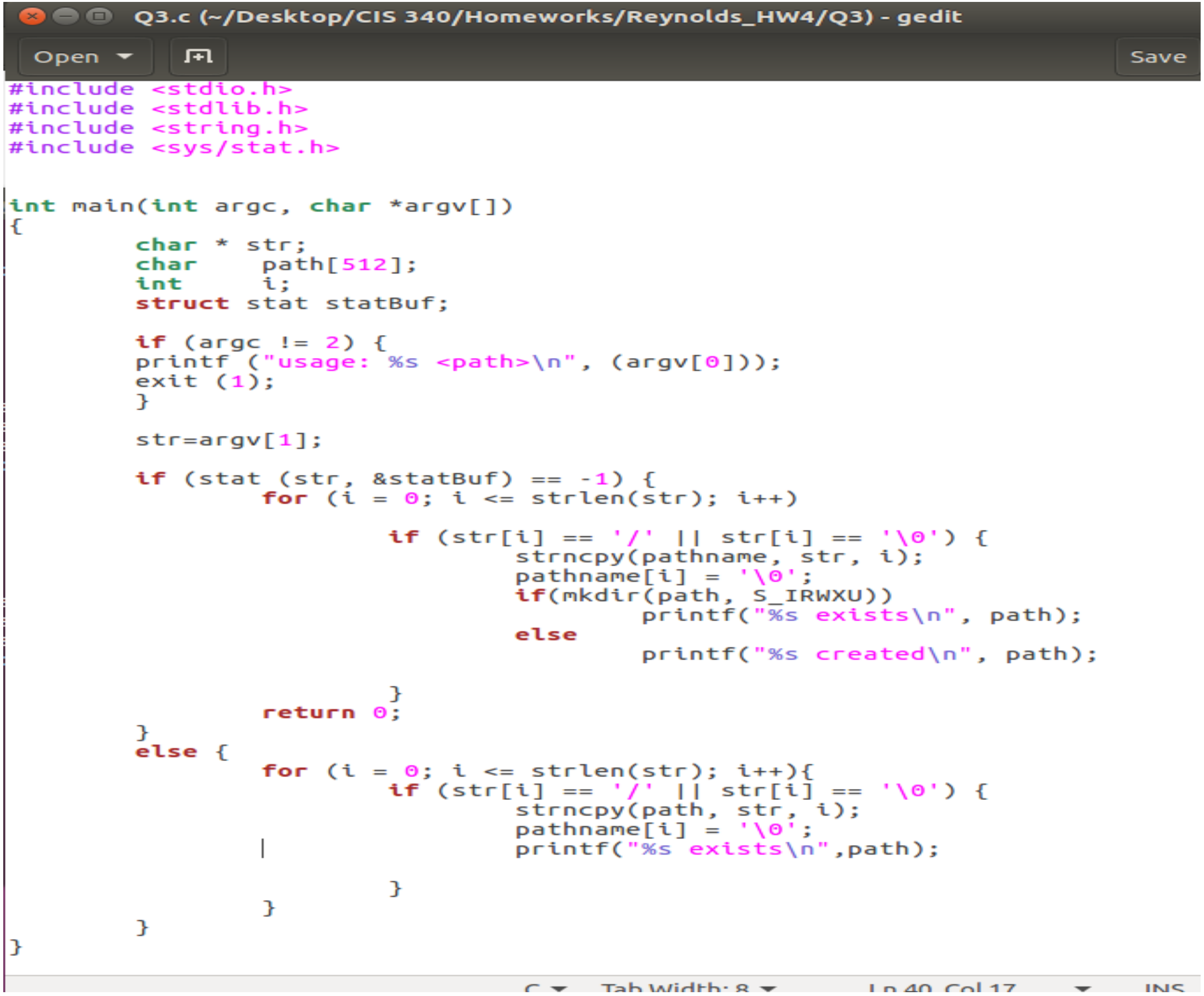


```c
#include <stdio.h>
int Hello();
int World();
main(){
        int test[5];
        test[0]=(int)Hello();
        test[1]=(int)Hello();
        test[2]=(int)World();
        test[3]=(int)World();
        test[4]=(int)Hello();
        printf("\nResults:");
        for(int i=0; i<5;i++){
                printf("%i ",test[i]);

        }

}
```

Question 3:

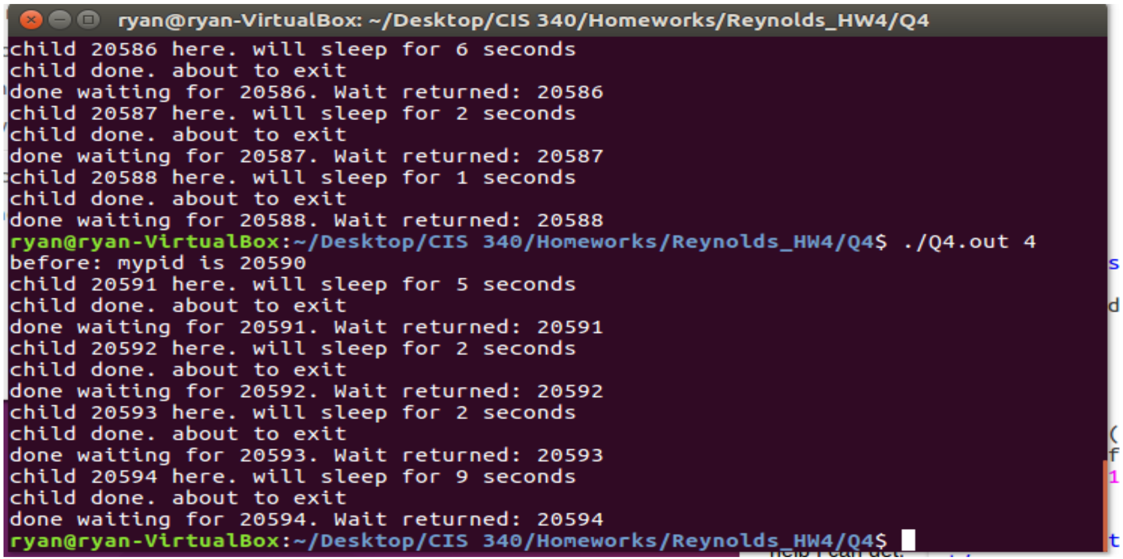

ryan@ryan-VirtualBox: ~/Desktop/CIS 340/Homeworks/Reynolds_HW4/Q3

```
ryan@ryan-VirtualBox:~/Desktop/CIS 340/Homeworks/Reynolds_HW4/Q3$ ./Q3.out /home/ryan/Desktop/"CIS 340"/Homeworks/Reynolds_HW4/Q3_test/Q3SubTest
 exists
/home exists
/home/ryan exists
/home/ryan/Desktop exists
/home/ryan/Desktop/CIS 340 exists
/home/ryan/Desktop/CIS 340/Homeworks exists
/home/ryan/Desktop/CIS 340/Homeworks/Reynolds_HW4 exists
/home/ryan/Desktop/CIS 340/Homeworks/Reynolds_HW4/Q3_test created
/home/ryan/Desktop/CIS 340/Homeworks/Reynolds_HW4/Q3_test/Q3SubTest created
ryan@ryan-VirtualBox:~/Desktop/CIS 340/Homeworks/Reynolds_HW4/Q3$ ./Q3.out /home/ryan/Desktop/"CIS 340"/Homeworks/Reynolds_HW4/Q3_test/Q3SubTest
 exists
/home exists
/home/ryan exists
/home/ryan/Desktop exists
/home/ryan/Desktop/CIS 340 exists
/home/ryan/Desktop/CIS 340/Homeworks exists
/home/ryan/Desktop/CIS 340/Homeworks/Reynolds_HW4 exists
/home/ryan/Desktop/CIS 340/Homeworks/Reynolds_HW4/Q3_test exists
/home/ryan/Desktop/CIS 340/Homeworks/Reynolds_HW4/Q3_test/Q3SubTest exists
ryan@ryan-VirtualBox:~/Desktop/CIS 340/Homeworks/Reynolds_HW4/Q3$ 
```



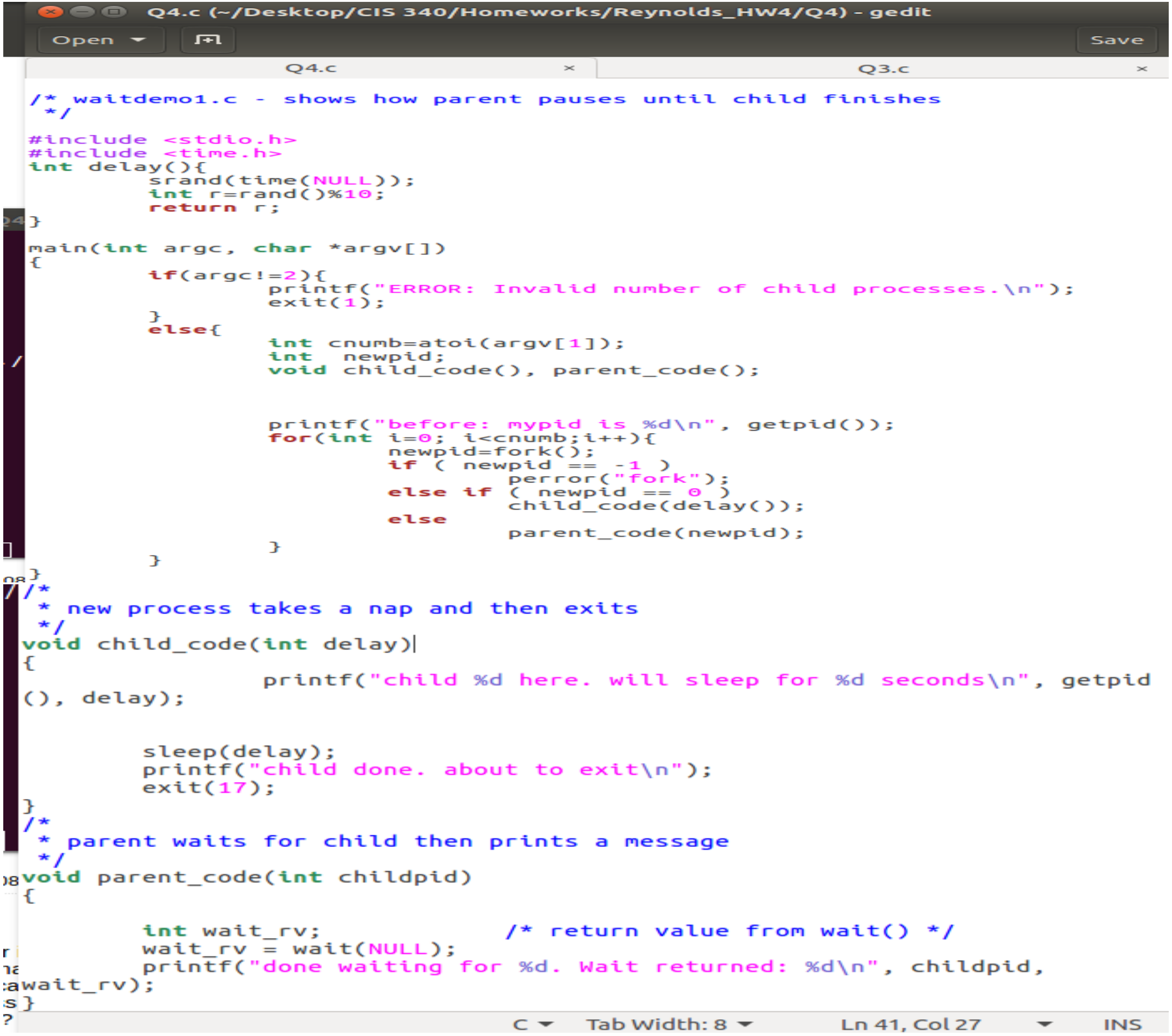
```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/stat.h>


int main(int argc, char *argv[])
{
        char * str;
        char    path[512];
        int     i;
        struct stat statBuf;

        if (argc != 2) {
        printf ("usage: %s <path>\n", (argv[0]));
        exit (1);
        }

        str=argv[1];

        if (stat (str, &statBuf) == -1) {
                for (i = 0; i <= strlen(str); i++)

                        if (str[i] == '/' || str[i] == '\0') {
                                strncpy(pathname, str, i);
                                pathname[i] = '\0';
                                if(mkdir(path, S_IRWXU))
                                        printf("%s exists\n", path);
                                else
                                        printf("%s created\n", path);

                        }
                return 0;
        }
        else {
                for (i = 0; i <= strlen(str); i++){
                        if (str[i] == '/' || str[i] == '\0') {
                                strncpy(path, str, i);
                                pathname[i] = '\0';
                                printf("%s exists\n",path);

                        }
                }
        }
}
```

Question 4:



ryan@ryan-VirtualBox: ~/Desktop/CIS 340/Homeworks/Reynolds_HW4/Q4

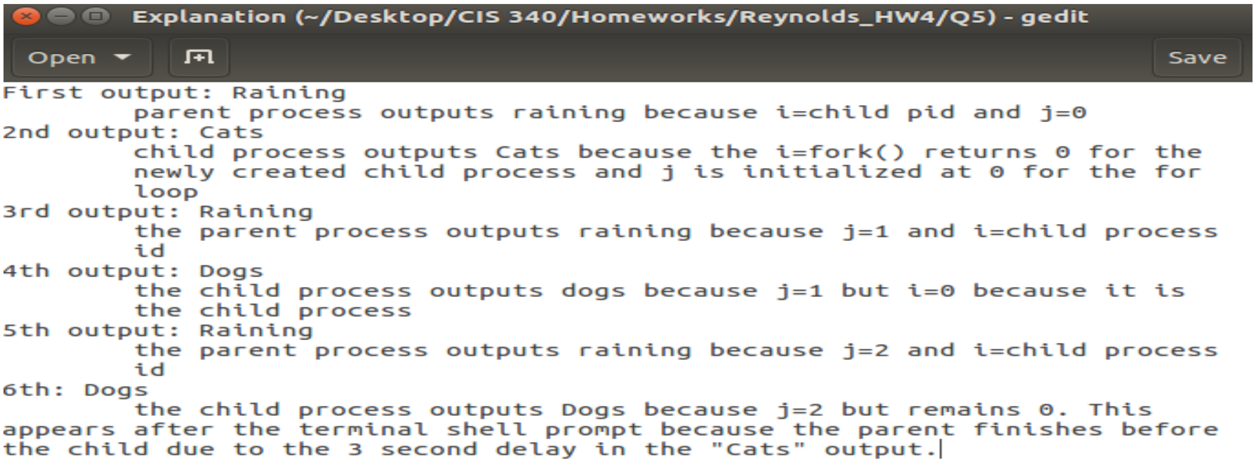
```
child 20586 here. will sleep for 6 seconds
child done. about to exit
done waiting for 20586. Wait returned: 20586
child 20587 here. will sleep for 2 seconds
child done. about to exit
done waiting for 20587. Wait returned: 20587
child 20588 here. will sleep for 1 seconds
child done. about to exit
done waiting for 20588. Wait returned: 20588
ryan@ryan-VirtualBox:~/Desktop/CIS 340/Homeworks/Reynolds_HW4/Q4$ ./Q4.out 4
before: mypid is 20590
child 20591 here. will sleep for 5 seconds
child done. about to exit
done waiting for 20591. Wait returned: 20591
child 20592 here. will sleep for 2 seconds
child done. about to exit
done waiting for 20592. Wait returned: 20592
child 20593 here. will sleep for 2 seconds
child done. about to exit
done waiting for 20593. Wait returned: 20593
child 20594 here. will sleep for 9 seconds
child done. about to exit
done waiting for 20594. Wait returned: 20594
ryan@ryan-VirtualBox:~/Desktop/CIS 340/Homeworks/Reynolds_HW4/Q4$ 
```



Q4.c (~/Desktop/CIS 340/Homeworks/Reynolds_HW4/Q4) - gedit

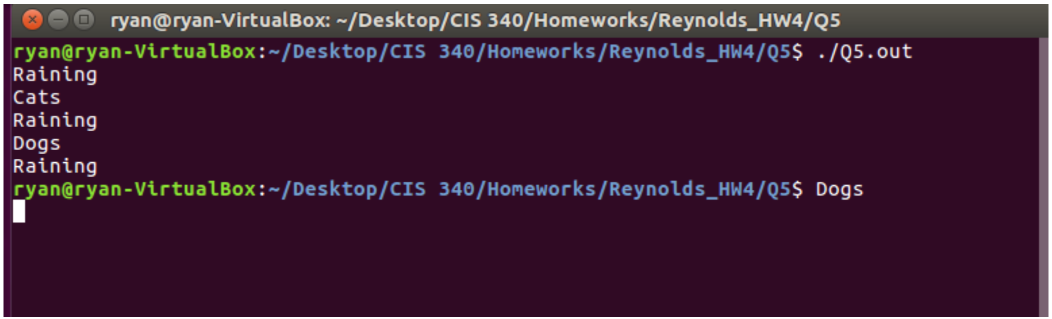
```
/* waitdemo1.c - shows how parent pauses until child finishes
 */
#include <stdio.h>
#include <time.h>
int delay(){
        srand(time(NULL));
        int r=rand()%10;
        return r;
}
main(int argc, char *argv[])
{
        if(argc!=2){
                printf("ERROR: Invalid number of child processes.\n");
                exit(1);
        }
        else{
                int cnumb=atoi(argv[1]);
                int  newpid;
                void child_code(), parent_code();


                printf("before: mypid is %d\n", getpid());
                for(int i=0; i<cnumb;i++){
                        newpid=fork();
                        if ( newpid == -1 )
                                perror("fork");
                        else if ( newpid == 0 )
                                child_code(delay());
                        else
                                parent_code(newpid);
                }
        }
}
/*
 * new process takes a nap and then exits
 */
void child_code(int delay)
{
                printf("child %d here. will sleep for %d seconds\n", getpid
(), delay);


        sleep(delay);
        printf("child done. about to exit\n");
        exit(17);
}
/*
 * parent waits for child then prints a message
 */
void parent_code(int childpid)
{

        int wait_rv;            /* return value from wait() */
        wait_rv = wait(NULL);
        printf("done waiting for %d. Wait returned: %d\n", childpid,
wait_rv);
}
```

Question 5:



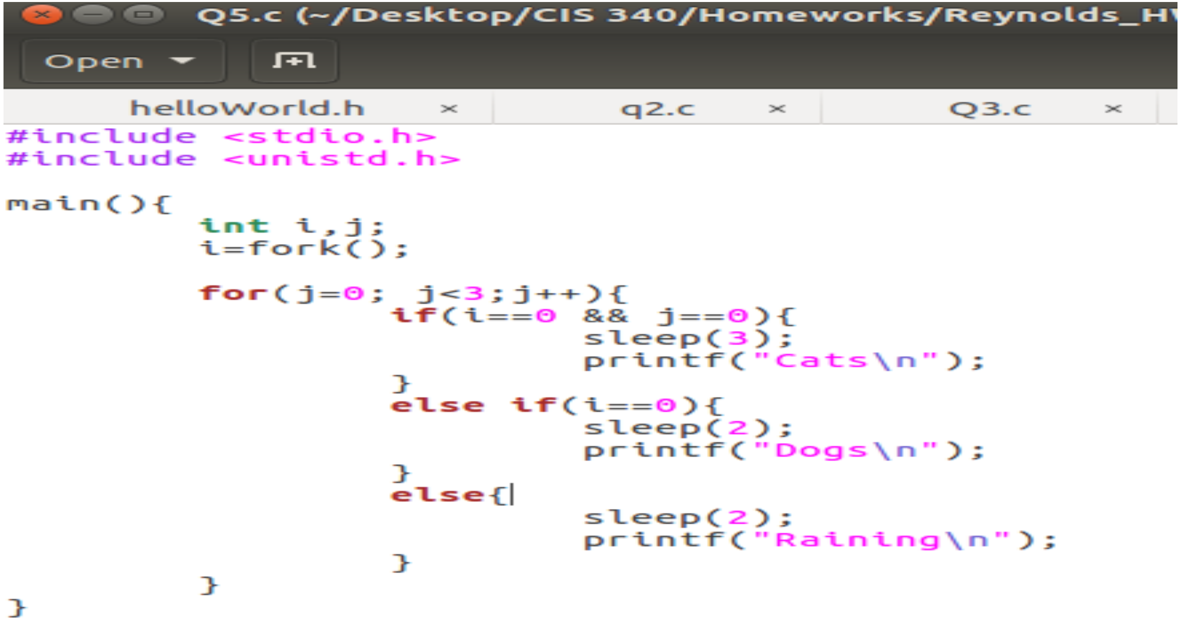
Explanation (~/Desktop/CIS 340/Homeworks/Reynolds_HW4/Q5) - gedit
Open
Save

```
First output: Raining
        parent process outputs raining because i=child pid and j=0
2nd output: Cats
        child process outputs Cats because the i=fork() returns 0 for the
        newly created child process and j is initialized at 0 for the for
        loop
3rd output: Raining
        the parent process outputs raining because j=1 and i=child process
        id
4th output: Dogs
        the child process outputs dogs because j=1 but i=0 because it is
        the child process
5th output: Raining
        the parent process outputs raining because j=2 and i=child process
        id
6th: Dogs
        the child process outputs Dogs because j=2 but remains 0. This
appears after the terminal shell prompt because the parent finishes before
the child due to the 3 second delay in the "Cats" output.
```



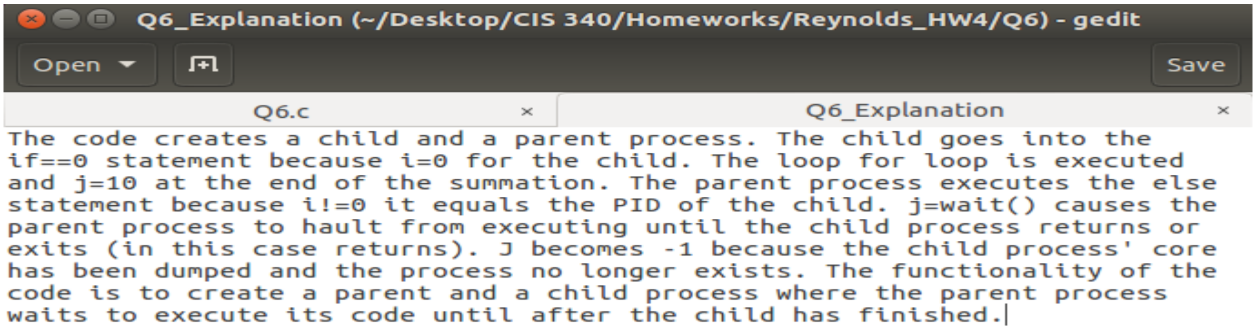
ryan@ryan-VirtualBox: ~/Desktop/CIS 340/Homeworks/Reynolds_HW4/Q5

```
ryan@ryan-VirtualBox:~/Desktop/CIS 340/Homeworks/Reynolds_HW4/Q5$ ./Q5.out
Raining
Cats
Raining
Dogs
Raining
ryan@ryan-VirtualBox:~/Desktop/CIS 340/Homeworks/Reynolds_HW4/Q5$ Dogs
```

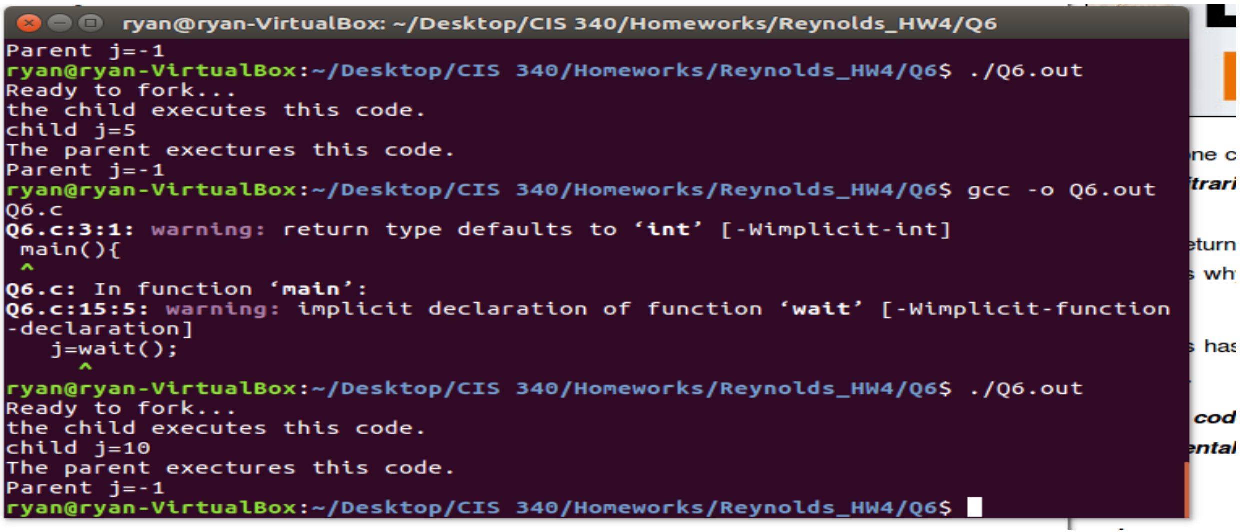


```
#include <stdio.h>
#include <unistd.h>

main(){
        int i,j;
        i=fork();

        for(j=0; j<3;j++){
                if(i==0 && j==0){
                        sleep(3);
                        printf("Cats\n");
                }
                else if(i==0){
                        sleep(2);
                        printf("Dogs\n");
                }
                else{
                        sleep(2);
                        printf("Raining\n");
                }
        }
}
```

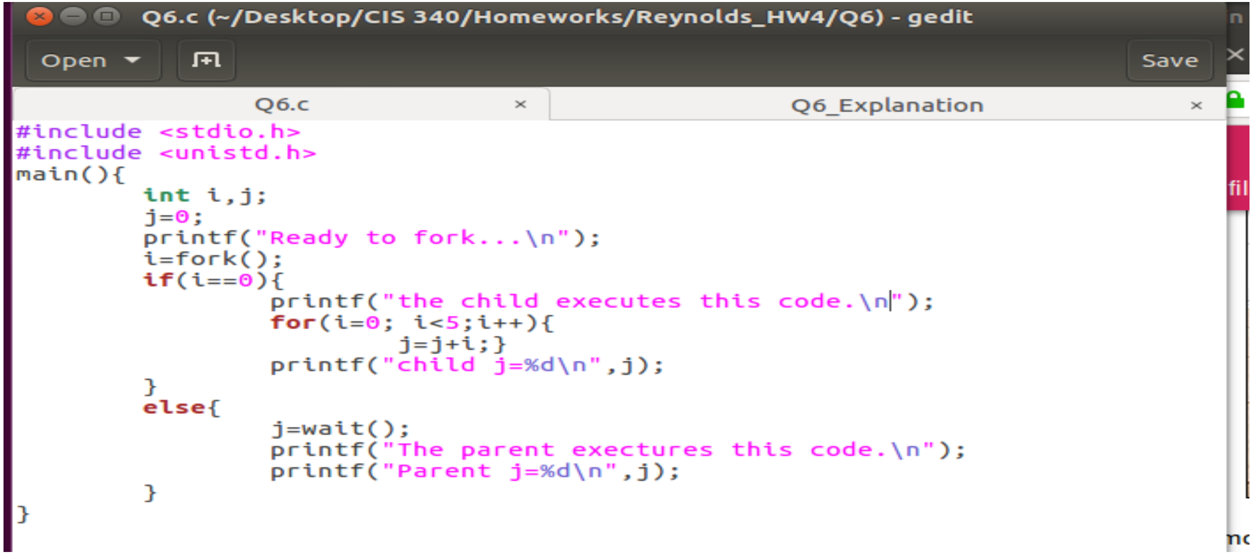
Question 6:


Q6_Explanation (~/Desktop/CIS 340/Homeworks/Reynolds_HW4/Q6) - gedit
Open
Save
Q6.c
Q6_Explanation

The code creates a child and a parent process. The child goes into the if==0 statement because i=0 for the child. The loop for loop is executed and j=10 at the end of the summation. The parent process executes the else statement because i!=0 it equals the PID of the child. j=wait() causes the parent process to hault from executing until the child process returns or exits (in this case returns). J becomes -1 because the child process' core has been dumped and the process no longer exists. The functionality of the code is to create a parent and a child process where the parent process waits to execute its code until after the child has finished.


ryan@ryan-VirtualBox: ~/Desktop/CIS 340/Homeworks/Reynolds_HW4/Q6

```
Parent j=-1
ryan@ryan-VirtualBox:~/Desktop/CIS 340/Homeworks/Reynolds_HW4/Q6$ ./Q6.out
Ready to fork...
the child executes this code.
child j=5
The parent exectures this code.
Parent j=-1
ryan@ryan-VirtualBox:~/Desktop/CIS 340/Homeworks/Reynolds_HW4/Q6$ gcc -o Q6.out
Q6.c
Q6.c:3:1: warning: return type defaults to ‘int’ [-Wimplicit-int]
 main(){
 ^
Q6.c: In function ‘main’:
Q6.c:15:5: warning: implicit declaration of function ‘wait’ [-Wimplicit-function
-declaration]
   j=wait();
     ^
ryan@ryan-VirtualBox:~/Desktop/CIS 340/Homeworks/Reynolds_HW4/Q6$ ./Q6.out
Ready to fork...
the child executes this code.
child j=10
The parent exectures this code.
Parent j=-1
ryan@ryan-VirtualBox:~/Desktop/CIS 340/Homeworks/Reynolds_HW4/Q6$
```



```
#include <stdio.h>
#include <unistd.h>
main(){
        int i,j;
        j=0;
        printf("Ready to fork...\n");
        i=fork();
        if(i==0){
                printf("the child executes this code.\n");
                for(i=0; i<5;i++){
                        j=j+i;}
                printf("child j=%d\n",j);
        }
        else{
                j=wait();
                printf("The parent exectures this code.\n");
                printf("Parent j=%d\n",j);
        }
}
```