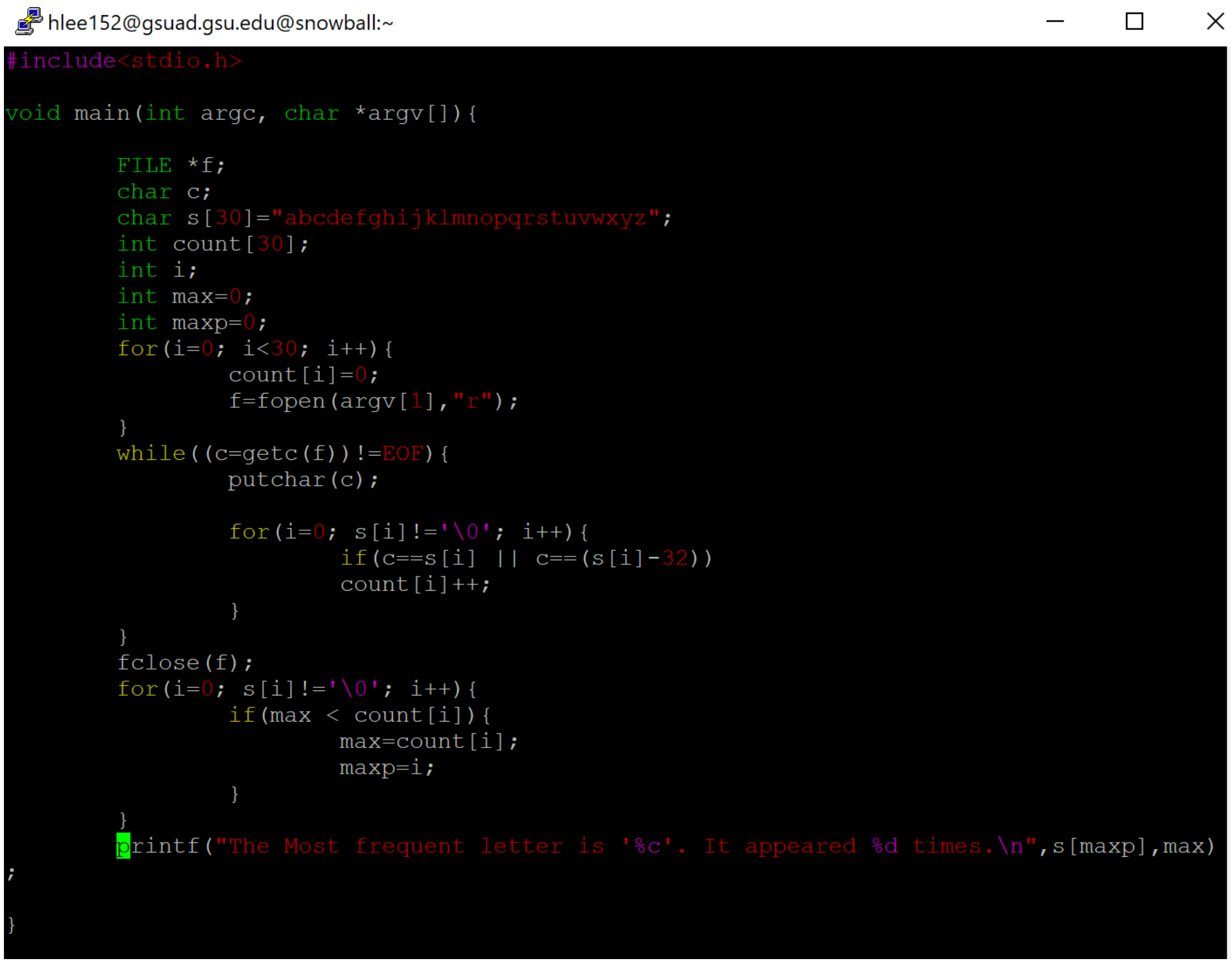
Lab9

Hyunki Lee

Part1

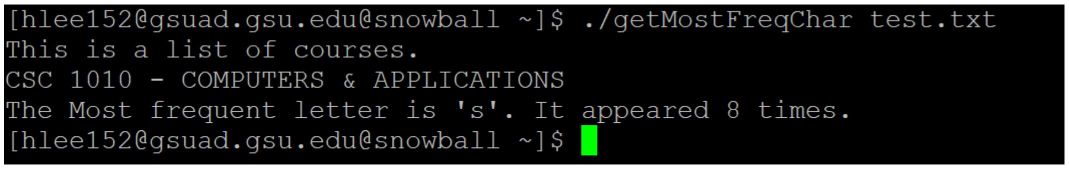
Code



hlee152@gsuad.gsu.edu@snowball:~

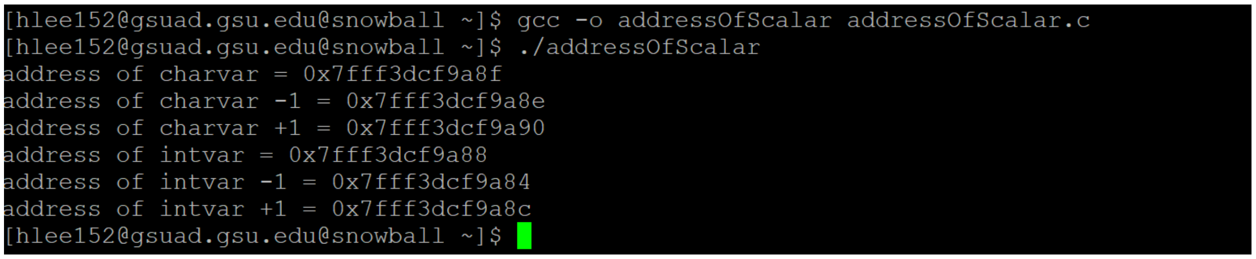
```c
#include<stdio.h>

void main(int argc, char *argv[]){

        FILE *f;
        char c;
        char s[30]="abcdefghijklmnopqrstuvwxyz";
        int count[30];
        int i;
        int max=0;
        int maxp=0;
        for(i=0; i<30; i++){
                count[i]=0;
                f=fopen(argv[1],"r");
        }
        while((c=getc(f))!=EOF){
                putchar(c);

                for(i=0; s[i]!='\0'; i++){
                        if(c==s[i] || c==(s[i]-32))
                        count[i]++;
                }
        }
        fclose(f);
        for(i=0; s[i]!='\0'; i++){
                if(max < count[i]){
                        max=count[i];
                        maxp=i;
                }
        }
        printf("The Most frequent letter is '%c'. It appeared %d times.\n",s[maxp],max)
;

}
```

Output

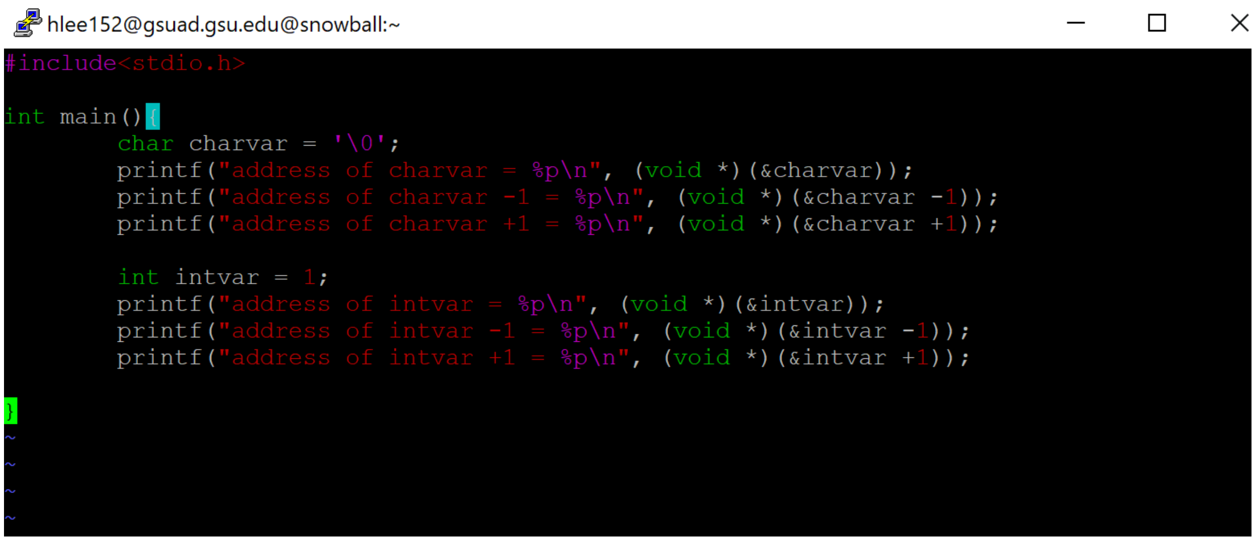

```
[hlee152@gsuad.gsu.edu@snowball ~]$ ./getMostFreqChar test.txt
This is a list of courses.
CSC 1010 - COMPUTERS & APPLICATIONS
The Most frequent letter is 's'. It appeared 8 times.
[hlee152@gsuad.gsu.edu@snowball ~]$
```

Part2

1) Output


```
[hlee152@gsuad.gsu.edu@snowball ~]$ gcc -o addressOfScalar addressOfScalar.c
[hlee152@gsuad.gsu.edu@snowball ~]$ ./addressOfScalar
address of charvar = 0x7fff3dcf9a8f
address of charvar -1 = 0x7fff3dcf9a8e
address of charvar +1 = 0x7fff3dcf9a90
address of intvar = 0x7fff3dcf9a88
address of intvar -1 = 0x7fff3dcf9a84
address of intvar +1 = 0x7fff3dcf9a8c
[hlee152@gsuad.gsu.edu@snowball ~]$
```

2) Code


hlee152@gsuad.gsu.edu@snowball:~

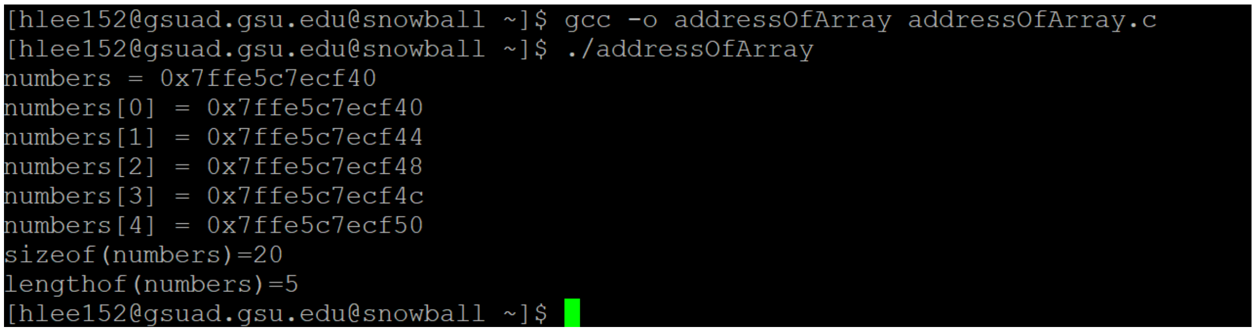
```
#include<stdio.h>

int main(){
        char charvar = '\0';
        printf("address of charvar = %p\n", (void *)(&charvar));
        printf("address of charvar -1 = %p\n", (void *)(&charvar -1));
        printf("address of charvar +1 = %p\n", (void *)(&charvar +1));

        int intvar = 1;
        printf("address of intvar = %p\n", (void *)(&intvar));
        printf("address of intvar -1 = %p\n", (void *)(&intvar -1));
        printf("address of intvar +1 = %p\n", (void *)(&intvar +1));

}
~
~
~
~
```

3)

Integer value is treated as 4bytes in C language. When we did any arithmetic for integer, the integer value will take 4 bytes of memory. Therefore, address for integer value also moves 4 bytes.

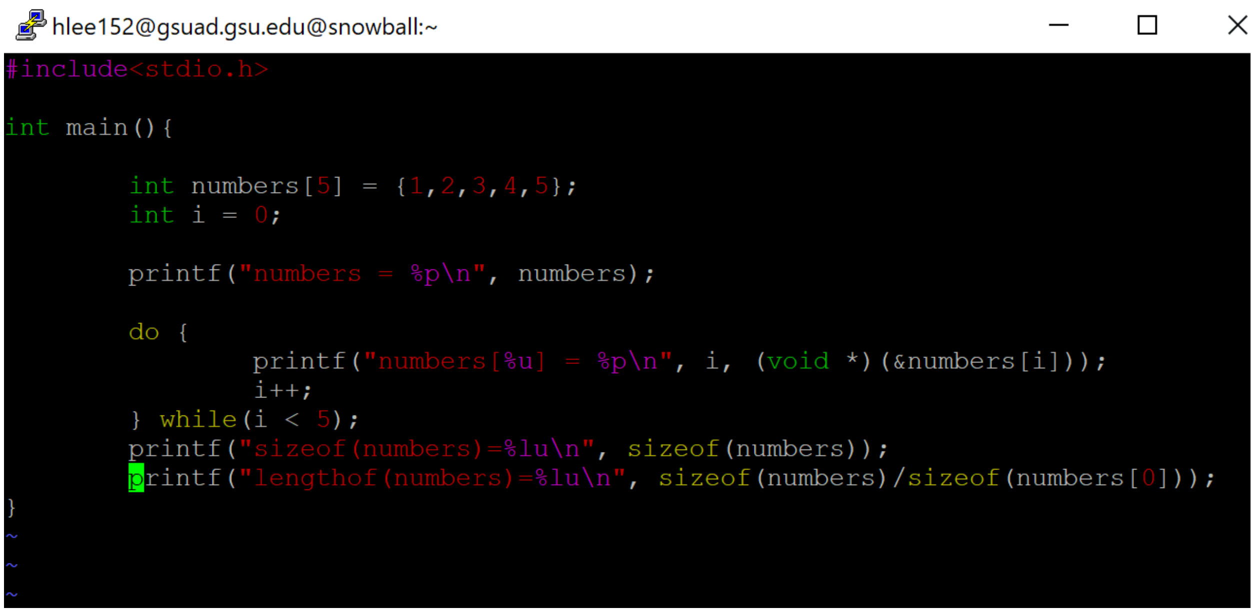
Part3

1) Output (include lengthof)



```
[hlee152@gsuad.gsu.edu@snowball ~]$ gcc -o addressOfArray addressOfArray.c
[hlee152@gsuad.gsu.edu@snowball ~]$ ./addressOfArray
numbers = 0x7ffe5c7ecf40
numbers[0] = 0x7ffe5c7ecf40
numbers[1] = 0x7ffe5c7ecf44
numbers[2] = 0x7ffe5c7ecf48
numbers[3] = 0x7ffe5c7ecf4c
numbers[4] = 0x7ffe5c7ecf50
sizeof(numbers)=20
lengthof(numbers)=5
[hlee152@gsuad.gsu.edu@snowball ~]$
```

Code (include lengthof)



hlee152@gsuad.gsu.edu@snowball:~

```
#include<stdio.h>

int main(){

        int numbers[5] = {1,2,3,4,5};
        int i = 0;

        printf("numbers = %p\n", numbers);

        do {
                printf("numbers[%u] = %p\n", i, (void *)(&numbers[i]));
                i++;
        } while(i < 5);
        printf("sizeof(numbers)=%lu\n", sizeof(numbers));
        printf("lengthof(numbers)=%lu\n", sizeof(numbers)/sizeof(numbers[0]));
}
~
~
~
```

2)

Yes, they are the same address.

3)

printf(“length(numbers)=%lu\n”, sizeof(numbers)/sizeof(numbers[0]));