

Lab 9 Joshua Griffith

PART 1:

```
joshua@joshua-VirtualBox:~/Labs/Lab9$ ./a.out test.txt
This is a list of courses.
CSC 1010 - COMPUTERS & APPLICATIONS

The Most frequent letter is 's'. It appeared 8 times.
```

PART 2:

Questions:

1) Run the C program, attach a screenshot of the output in the answer sheet.

```
joshua@joshua-VirtualBox:~/Labs/Lab9$ ./a.out
address of charvar = 0x7ffde481fba3
address of charvar -1 = 0x7ffde481fba2
address of charvar +1 = 0x7ffde481fba4
address of intvar = 0x7ffde481fba4
address of intvar -1 = 0x7ffde481fba0
address of intvar +1 = 0x7ffde481fba8
```

2) Attach the source code in the answer sheet

```
#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));
```

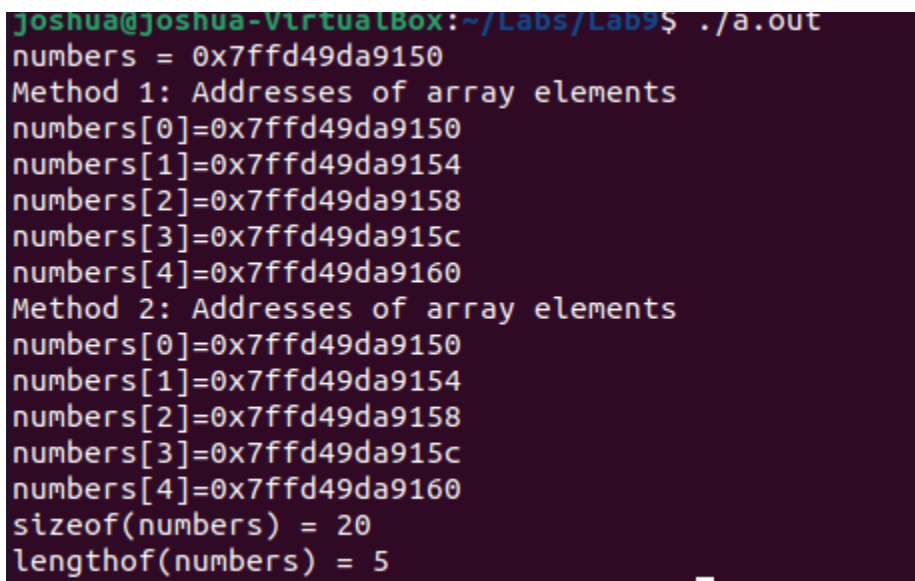
```
printf("address of intvar +1 = %p\n", (void*)&intvar+1));  
}
```

3) Then explain why the address after intvar is incremented by 4 bytes instead of 1 byte.

Since invar is declared as an integer, and integers take up 4 bytes of memory whereas charvar increase by 1 byte as it's declared as a char

PART 3:

1)

A terminal window with a dark purple background and light green text. The prompt is 'joshua@joshua-VirtualBox: ~/Labs/Lab9\$'. The command './a.out' has been executed. The output shows the memory addresses of array elements 'numbers' from index 0 to 4, using two different methods. It also shows the total size of the array in bytes (20) and the number of elements (5).

```
joshua@joshua-VirtualBox: ~/Labs/Lab9$ ./a.out  
numbers = 0x7ffd49da9150  
Method 1: Addresses of array elements  
numbers[0]=0x7ffd49da9150  
numbers[1]=0x7ffd49da9154  
numbers[2]=0x7ffd49da9158  
numbers[3]=0x7ffd49da915c  
numbers[4]=0x7ffd49da9160  
Method 2: Addresses of array elements  
numbers[0]=0x7ffd49da9150  
numbers[1]=0x7ffd49da9154  
numbers[2]=0x7ffd49da9158  
numbers[3]=0x7ffd49da915c  
numbers[4]=0x7ffd49da9160  
sizeof(numbers) = 20  
lengthof(numbers) = 5
```

2) Check the address of the array and the address of the first element in the array. Are they the same?

They are the same since the address of an array is the address of the first element

3) Write down the statement to print out the length of the array by using sizeof operator.

```
printf("Length(numbers) = %lu\n", sizeof(numbers)/sizeof(numbers[0]));
```