Lab 3: Introduction to Data Types and I/O Matthew Szymanski 4/21/2021

1. **Write a C program named "sizeDataTypes.c" that prints out the size of the following data types: *short*, *int*, *char*, *double*, *long*, and *long double*. This can be done by using the *sizeof* operator. Also show that the following Boolean checks are true.**

(4 points)

printf("%d\n", sizeof(short) < sizeof(int));
printf("%d\n", sizeof(char) < sizeof(short));
printf("%d\n", sizeof(long) > sizeof(int));
printf("%d\n", sizeof(long double) > sizeof(double));

```
mszymanski@DESKTOP-RD9BRN8:~/Lab3$ ./sizeDataTypes
Size of short is: 2
Size of int is: 4
Size of char is: 1
Size of double is: 8
Size of long is: 8
Size of long double is: 16
1
1
1
1
mszymanski@DESKTOP-RD9BRN8:~/Lab3$
```

mszymanski@DESKTOP-RD9BRN8: ~/Lab3

GNU nano 2.9.3 sizeDataTypes.c

```
#include <stdio.h>
int main(){
        short sInt = 20;
        int oInt = 100;
        char charType;
        double d = 10.00;
        long lVar = 1000000;
        long double ldVar = 2000000000;
printf("Size of short is: %zu\n", sizeof(sInt));
printf("Size of int is: %zu\n", sizeof(oInt));
printf("Size of char is: %zu\n", sizeof(charType));
printf("Size of double is: %zu\n", sizeof(d));
printf("Size of long is: %zu\n", sizeof(lVar));
printf("Size of long double is: %zu\n", sizeof(ldVar));


printf("%d\n", sizeof(short) < sizeof(int));
printf("%d\n", sizeof(char) < sizeof(short));
printf("%d\n", sizeof(long) > sizeof(int));
printf("%d\n", sizeof(long double) > sizeof(double));



return 0;
}
```

2. **From the following code snippet, what will be the final values of the variables '*a*', '*b*', '*c*', and '*size*'. Write a program named 'finalValues.c' that prints the values of these 4(four) variables. (4 points)**

//file name:
finalValues.c int a = 3;
int b = 4; double c = ++a
+ b++; size_t size =
sizeof(c/a);

```
mszymanski@DESKTOP-RD9BRN8:~/Lab3$ ./finalValues
4
5
8.000000
8
mszymanski@DESKTOP-RD9BRN8:~/Lab3$
```

mszymanski@DESKTOP-RD9BRN8: ~/Lab3

GNU nano 2.9.3 finalValues.c

```c
#include <stdio.h>
int main(){
//file name:
int a = 3;
int b = 4;
double c = ++a + b++;
size_t size = sizeof(c/a);
printf("%u\n", a);
printf("%u\n", b);
printf("%f\n", c);
printf("%zu\n", size);

return 0;
}
```

3. **Consider the following program that asks the user for 10 positive integers (grade).**
(6 points total)

```
#include <stdio.h> //file Name: AverageGrade.c

int main(){
	int i = 0;
	int grade;

	printf("Please input 10 grades :\n");

	while(i < 10) {
	scanf("%d", &grade);
		i++;
	}
	return 0;
}
```

Modify the program as follows:

a. The program will be able to read any number of inputs from the keyboard. -1000 will indicate the end of processing. **(2 points)**

b. As the user keeps on typing integers, the program will keep on summing and averaging the numbers typed so far. For example, in the following screenshot, the user typed 10, so, the sum and average are both 10. Next, the user typed 20, the sum and average were calculated considering the previous input '10'. This will continue until the user types -1000. You need to find the sum and average in double.

(4 points)

```
syasmin@cscd-linux01:~/CPractice$ ./AverageGrade
Please input a number of grades:
10
Sum is 10.000000
Average is 10.000000
20
Sum is 30.000000
Average is 15.000000
45
Sum is 75.000000
Average is 25.000000
56
Sum is 131.000000
Average is 32.750000
-1000
syasmin@cscd-linux01:~/CPractice$
```

```
mszymanski@DESKTOP-RD9BRN8:~/Lab3$ ./Avera
Please input 10 grades :
10
Sum is 10.000000
Average is 10.000000
20
Sum is 30.000000
Average is 15.000000
30
Sum is 60.000000
Average is 20.000000
45
Sum is 105.000000
Average is 26.250000
-1000
Sum is -895.000000
Average is -179.000000
mszymanski@DESKTOP-RD9BRN8:~/Lab3$
```

4. Answer the following question about getchar. **(4 points total)** a. What is the problem with the following C program? **(2 points)**

b. How will you fix it? **(2 points)**

```
#include <stdio.h> //file name: charBug.c
 int
main(){
        int c;
        char dummy[10];

        printf("Enter a value :");
        c = getchar( );

        while(c != 'q' && c != 'Q'){
                printf("You entered: ");
                putchar(c);
        printf("\n");
                printf("Enter a value :");
                c = getchar();
        }
        return 0;
}
```

A) The problem with the following C program is that it does not display what the user entered like it says it will. Instead, it prints out…

Has to deal with new line.

```
mszymanski@DESKTOP-RD9BRN8:~/Lab3$ ./charBug
Enter a value :1
You entered: 1
Enter a value :You entered:

Enter a value :q
mszymanski@DESKTOP-RD9BRN8:~/Lab3$ ./charBug
```

B)

```
mszymanski@DESKTOP-RD9BRN8:~/Lab3$ nano charBug.c
mszymanski@DESKTOP-RD9BRN8:~/Lab3$ gcc -o charBug charBug.c
mszymanski@DESKTOP-RD9BRN8:~/Lab3$ ./charBug
Enter a value :a
You entered: a
Enter a value :b
You entered: b
Enter a value :c
You entered: c
Enter a value :q
mszymanski@DESKTOP-RD9BRN8:~/Lab3$
```

5. Answer the following questions on gets and fgets. **(4 points total)**

a. What are the differences between 'gets' and 'fgets' functions? **(2 points)**

b. Next, rewrite the following code with fgets. **(2 points)**

```
#include <stdio.h>                    //file name: getsFgets.c
int main() {
        char name[100];
        printf("Please enter a name:");
        gets(name);
        printf("The name you entered is %s", name);
        return 0;
}
```

a) Gets takes one parameter and replaces new line character with null character. fgets operates similarly to gets but, preserves the new line character and appends a null character '\0' on the end.

```
mszymanski@DESKTOP-RD9BRN8:~/Lab3$ ./g
Please enter a name:Matthew
The name you entered is Matthew
mszymanski@DESKTOP-RD9BRN8:~/Lab3$
```

6. Write a C program that takes user input in the form of a sentence from the keyboard. The user types the sentence from the keyboard and presses the 'Enter' key. Next, the user is asked to type a character (this is an alphabetic character). The program then displays the number of times the character (that was typed by the user) shows up in the sentence. This must ignore cases (i.e., 'A' and 'a' will be considered as the same letter regardless of the upper case or lower case.). The following example uses the sentence "Hello world" and the character to be found is 'O'. The program will generate the following output.

(8 points)

```
Toastmaker@MSI:/mnt/d/CScD Class/CScD 240 AT/19SP/Lab3$ ./CC
Type a sentence:
Hello World
Type a character that you'd like to find the number of occurrences in a sentence:
O
Alphabet O has a frequency 2
Toastmaker@MSI:/mnt/d/CScD Class/CScD 240 AT/19SP/Lab3$
```

Consider the following 'C' code "countChar.c". The comments in code show the steps you need to write to make the code work to answer Question 6.

```
#include <stdio.h>
#define MAXSIZE 100

int main(){
        char ch;
        int i;
        char sentence[MAXSIZE];    int
alphabetsCount = 0;
        printf("Type a sentence:
\n");
        // write the code that reads a sentence from the keyboard            2pts
        printf("Type a character that you'd like to find the number of occurrences in a
sentence:\n");
        // write the code that reads a character from the keyboard           2 pts
        // write the code that counts the number of occurrences of the
        // character in sentence;                                            2 pts

        // ignore uppercase or lowercase                                     2
pts     printf("Alphabet %c has a frequency of %d\n", ch, alphabetsCount);
        return 0;
}
```

```
mszymanski@DESKTOP-RD9BRN8:~/Lab3$ ./countChar
Type a sentence:
Hello World
Type a character that you'd like to find the number of occurrences in a sentence:
o
Alphabet o has a frequency of 2
mszymanski@DESKTOP-RD9BRN8:~/Lab3$
```

Submission:

- All answers (whenever you need to write codes) should be submitted as complete C codes named as **sizeDataTypes.c (Question 1)**, **finalValues.c (Question 2) AverageGrade.c (Question 3), charBug.c (Question 4b)**, **getsFgets.c (Question 5b),** and **countChar.c (Question 6)** respectively.

- **Question 4a and 5a** should be submitted in a PDF file named **Lab3.pdf**.
- Full submission should contain all **C** files and **pdf** file zipped as follows: your last name, first letter of your first name, Lab3.pdf (i.e., YasminSLab3.zip). You should turn in through the EWU Canvas system.

- Submission deadline is **Friday, April 2**3.
- **No late submission will be accepted**.