202018021
Fu, Ziyu
Programming I Final Report
2021/07/07

Part 1

1-1

This function would take an uppercase letter in the `US-ASCII` standard and convert it into the corresponding lowercase letter.

1-2

Program:

```
#include <stdio.h>

char utol(char c){
   return (c+0x20);
}

int main(){

   printf("upper: %c, lower: %c\n", 'A', utol('A'));

   return 0;
}
```

Output:

```
upper: A, lower: a
```

Part 2

```
void get_min_student(int numberOfPeople, float grades[]){

   float min = 101;

   for (int i = 0; i < numberOfPeople; ++i){
       if (grades[i] < min){
           min = grades[i];
       }
   }

   for (int i = 0; i < numberOfPeople; ++i){
       if (grades[i] == min){
           printf("%d\n", i);
       }
   }

}
```

Part 3

More than one kind of data type can satisfy this expected input range, as long as it satisfies the following criterias.

1) As this method of calculation is only possible because of truncated integers, the data type of **`bunshi`** and **`bunbo`** has to be **`integer`**.
2) Because the expected input range is 1 to 65535, both **`bunshi`** and **`bunbo`** must have at least 2 bytes allocated to the variable, and
3) If only 2 bytes are being used, it must be an **`unsigned`** integer type.

These are the only restrictions for this particular calculation to work. So any integer data types like **`short`**, **`int`**, or **`long`**, etc. would work in this case.

However, the **`scanf()`** function in the sample code has specific string formatting in place. The input for **`bunshi`** expects a **`%d`**, which is an **`int`** type, and the input for bunbo is **`%hd`**, which corresponds to a **`short`** type.

In this case, since the size for **`short`** is only 2 bytes, **`bunbo`** must be an **`unsigned short`**. And this calculation does not behave consistently with negative outputs, so it is necessary to restrict **`bunshi`** to an **`unsigned int`** as well.

In conclusion, in order to correctly fill in the blank for this specific question:

A needs to be an **`unsigned int`** type.
B needs to be an **`unsigned short`** type.

In reality, **`int`** would satisfy the need for both A and B. Modern computers won't run into memory shortages just for this, and most C compilers can implicitly convert data types anyway.

Part 4

Change `int flag = 1;` to `static int flag = 1;` on **`line 13`**.

`static` variables preserve “static” memory allocations, so its value won’t get erased with each call.

Part 5

5-1

```
void plot(void){
   // plot
   for (x = 0; x < SIZE_X; x++){
       y = Offset + Offset * sin(PIE/2 + 2*(2.0 * PIE * x / SIZE_X));
       square[x][y] = '*';
   }
}
```

Double the frequency and displace the sine wave by pi/2 to get the defined output.

5-2

```
void plot(void){
   // plot
   for (x = 0; x < SIZE_X; x++){
       y = Offset + Offset * sin(2.0 * PIE * x / SIZE_X);
       square[x][y] = '*';
       y = 0.5 * (x) - 1;
       square[x][y] = '+';
   }
}
```

Add another equation that plots on top of the existing graph.