EE 285 Name____sec________________

Exam 2 – Dec. 16, 2016

Problem 1 (25 points) —

Here is a linear array of integers— a[8] = {3, 6, 9, 4, 2, 12, 16, 1};

a)
```
double someFunction( int a, int b, int c, int d ){

     double burp
     burp = (a + b + c + d)/4.0;
     return burp;
}

numb = someFunction( a[1], a[3], a[5], a[7] );
```

What value of numb is returned?

b)
```
void someFunction( int a_ray[] ){

   int i;

   // Be careful below.  Note the type of division at each step.
   // It is advisable to write the entire array at each step.
   a_ray[0] = ( a_ray[7] + a_ray[0] + a_ray[1] )/ 3;

   for( i = 1; i < 7; i++){
      a_ray[i] = ( a_ray[i-1] + a_ray[i] + a_ray[i+1] )/3;
    }

   a_ray[7] = ( a_ray[6] + a_ray[7] + a_ray[0] )/ 3;
}

someFunction( a );
```

What are the values in the elements of a after the function is called?

c)
```
void someFunction( int* arayPtr ){

   int i, temp;

   for( i = 0; i < 4; i++){
      temp = *(arayPtr + i);
      *(arayPtr + i) = *(arayPtr + 7 - i);
      *(arayPtr + 7 - i) = temp;
    }
}

someFunction( a );
```

What are the values in the elements of a after the function is called?

Problem 2 (25 points) — Write a C program that does the following:

1. Reads in the contents of a text file — inputFile.txt, shown below. Note that the first value in the file gives the number of lines of data — use this in reading the rest of the file.
2. Each line corresponds to one leg of a trip. The first item is a character that gives the direction of travel, the second item is a floating point number that gives the distance traveled, the third item is a integer that gives the time traveled, in minutes.
3. Read this data into an array of structs. Be sure to define the structs first, and then declare the array, called `tripLeg[]`. Each line of the file will correspond to one element of the struct array. The struct should have members: `direction`, `distance`, `time`, and `speed`.
4. Calculate the speed in miles per hour for each leg of the trip and store the value in the appropriate struct member.
5. Then calculate the total distance traveled, the total time traveled, and the average speed for the entire trip. Note that you will need variables for this quantities.
6. Finally, write the struct information, (direction, distance, time, and speed) to a new file — outputFile.txt. Also, at the end of the file, write the total distance, total time, and average speed.

Include whatever other variables are needed to do the job.

Contents of inputFile.txt

```
6
E 22.4 32
N 17.6 28
E 31.0 50
N 44.9 71
W 10.2 15
S  4.7  8
```

Problem 3 (25 points) — Write an Arduino program that does the following:

An Arduino has the following components attached — an on/off type light sensor connected to digital pin 4 (a digital input), a potentiometer is attached to analog pin 0 (an analog input, obviously), and 5 red LEDs that are connected to digital pins 3 – 7. The light sensor gives a HIGH voltage when it is light out and LOW voltage when it is dark. The potentiometer voltage varies between 0 V and 5 V depending on the pot setting. You can assume that the LEDs are connected to the digital pins with proper current-limiting resistors.

Write an Arduino program that:

1. When the room lights are out (i.e. when the light sensor is producing a LOW voltage) will turn on some of the LEDs, with the number of LEDs turned on depending on the potentiometer setting.
2. If $v_{pot} < 0.5$ V, no LEDs are turned on.
   If $0.5 \text{ v} \leq v_{pot} < 1.5$ V, one LED (connected to pin 3) is turned on.
   If $1.5 \text{ V} \leq v_{pot} < 2.5$ V, two LEDs (connected to pins 3 & 4) are turned on.
   If $2.5 \text{ V} \leq v_{pot} < 3.5$ V, three LEDs (connected to pins 3, 4, & 5) are turned on.
   If $3.5 \text{ V} \leq v_{pot} < 4.5$ V, four LEDs (connected to pins 3, 4, 5, & 6) are turned on.
   if If $v_{pot} \geq 4.5$ V, all five LEDs are turned on.
3. The potentiometer voltage is sampled and the LEDs are adjusted every 0.5 seconds. If you want, you can use the delay function for timing.

Include whatever variables are needed to do the job.

EE 285 exam “help sheet”

integer — `int anInteger;`
floating point — `double theDooble;`
character — `char c;` —> character literals are denoted with single quotes. For example: `'g'`, `'T'`.

linear array — `int linArray[20];`
2D array — `double bigArray[10][5]` (10 rows, 5 columns)

`%` —> Modulus operator. Returns the remainder of a division. Example: `23%4` = 3.

```
printf( “int = %d and floating point = %lf”, anInteger, theDooble );
```

`scanf( “%d”, &anInteger );` —> reads an integer from the keyboard.
`scanf( “%lf”, &theDooble);` —> reads an double length floating point from the keyboard.

>= —> greater than or equal to `( topValue >= maxValue )`
<= —> less than or equal to `( myAge <= 100 )`
== —> is equal to? `( x == 16)` double equals signs are an easily overlooked error.

`&&` —> logical AND, `( a > 0 && a <= 7)`

`||` —> logical OR, `( a >= MAX_VALUE || c == ‘Q’)`

`i++` —> increment by 1. Equivalent to `i = i + 1`. Equivalent to `++i`.
`i--` —> decrement by 1. Equivalent to `i = i - 1`. Equivalent to `--i` .

`a += delta` —> add `delta` to variable a. (Equivalent to `a = a + delta`.)
Same for `-, *, /`.

```
if( conditional statement ) {

    Do some stuff;
}
else {

    Instead do the other stuff;
}
```

The `else` part is optional (Note: if only one statement follows the if or else, then the braces are optional.)

```
while( conditional statement ){

      Do some stuff;
}

for( i = MIN; i <= MAX; i++) {

      Do some stuff;
}
```

(Again, if only one statement follows the `while` or `for`, then the braces are optional.) With the while loop, make sure that something is changing within the loop so that the conditional will become false at some point. (Avoid infinite loops.)

```
if( a == 1){
   printf(“%d: Get ready.”, a);
}
else if (a == 2){
   printf(“%d: Get set.”, a);
}
else if (a == 3){
   printf(“%d: Go!”, a);
}
```

```
switch( a ){

   case 1:
   printf(“%d: Get ready.”, a);
   break;

   case 2:
   printf(“%d: Get set.”, a);
   break;

   case 3:
   printf(“%d: Go!”, a);
   break;
}
```

Function declaration
Note: depending on how you have ordered the functions in your program file, you may need to include a prototype declaration before using the function. Here is an example of a function definition.

```
int bigger( int a, int b){

    if( a >= b){
        printf( “%d is bigger!”, a );
         return a;
    }
    else {
        printf( “No! %d is bigger!”, b );
         return b;
    }
}
```

```
struct structureTag {
   int member1;
   double member2;
   char member3;
   …
};
```

To declare an array of struct variables

```
struct structureTag structArray[n]
```

```
FILE* filePtr;

filePtr = fopen( “fileNameOnDisk.wtf”, “w” );
(“w” for write “r” for read.)

fclose( filePtr );

fprintf( filePtr, “format_string”, variables );

fscanf( filePtr, “format_string”, variables );
```

Arduino

Recall the it needs setup() and loop() functions.

Commands that might be useful:

```
• pinmode( pin, mode);         //modes are OUTPUT, INPUT, or INPUT_PULLUP
• digitalWrite( pin, output);  //output are HIGH or LOW
• input = digitalRead( pin );  //inputs can be either HIGH or LOW
• delay( time );               //delay for the given number of milliseconds
• input = analogRead( pin );   //0 - 5V at pin corresponds to 0-1023 for input
```