10.1. Write a short description of each of the following functions:

isalpha - accepts an integer argument `c` and returns an integer; tests to see if either `isupper` or `islower` is true for the argument, which is by convention expected to be limited to a `signed char` in width. Returns `isupper` or `islower`.
isalnum - accepts an integer argument `c` and tests whether `isalpha` or `isdigit` is true. `c` is expected to be a `signed char` in width. Returns true or false in integer width.
isdigit - Tells whether something is a numeric digit.
islower - Tells whether a character is a lower-case letter.
isprint - Tells whether a character is a printable character, which roughly means not a control character.
ispunct - Tells whether a character is punctuation.
isupper - Tells whether a character is an upper-case letter.
isspace - Tests against whitespace, to spaces, tabs, carriage returns, linefeeds, &c.
toupper - Takes a character and returns its upper-case variant iff it was a lower-case letter.
tolower - Takes a character and returns its lower-case variant iff it was an upper-case letter.

10.2. Write a statement that will convert the contents of the `char` variable `big` to lowercase. The converted value should be assigned to the variable `little`.

```
char little = tolower(big);
```

10.3. Write an `if` statement that will display the word “digit” if the variable `ch` contains a numeric digit. Otherwise, it should display “Not a digit.”

```
if(isdigit(ch)) {
      cout << “digit”;
} else {
      cout << “Not a digit.”;
}
```

10.4. What is the output of the following statement?

```
cout << toupper(tolower(‘A’));
```

It should print out A.

10.5. Write a loop that asks the user “`Do you want to repeat the program or quit? (R/Q)`”. The loop should repeat until the user has entered an R or Q (either uppercase or lowercase).

```
char input;
while(true) {
      cout << “Do you want to repeat the program or quit? (R/Q)”
      << endl;
      input = cin.get();
      if (tolower(input)==‘Q’) {
```

```
            break;
        }
    }
```

10.6. Write a short description of the following functions:

`strlen` - accepts a single argument of type `const char*` and counts characters from the first until it reaches a null terminator, and returns the count.
`strcat` - accepts two arguments of type `const char*` and appends characters from the first onto the second until a null-terminator is reached in the second. Note that this is inherently unsafe (see `strncat`).
`strcpy` - accepts two arguments of type `const char*` treated as destination and source, respectively, and copies bytes from source into destination until a null-terminator is found in source. The address for destination is then returned.
`strncat` - accepts three arguments, the first two of type `const char*` and the final of type `size_t`; and appends not more than the third argument (the `size_t`-typed argument) characters from the first onto the second until or null-terminator is reached in the second, whichever condition is satisfied first.
`strncpy` - accepts three arguments, two of type `const char*` as destination and source, respectively, and a final argument of type `size_t` as length; and copies no more than length bytes from source into destination, or until a null-terminator is found in source, whichever condition is met first. The address for destination is then returned.
`strcmp` - accepts two arguments of type `const char*` and compares each byte in both, returning 1 if the two arguments are identical up to and including their null-terminators, and returning `0` if and only if there is a discrepancy between the two.
`strstr` - accepts two arguments of type `const char*` as haystack and needle, and finds the first occurrence of the null-terminated needle in haystack and returns the location of the first character of needle in haystack, or −1 if needle is not present in haystack.

10.8. What will the following program segment display?

```
char string1[16] = “Have a “;
char string2[9] = “nice day”;
strcat(string1, string2);
cout << string1 << endl;
cout << string2 << endl;
```

This should output something to the following effect:

```
Have a nice day
nice day
```

10.9. Write a statement that will copy the string “Beethoven” to the array `composer`.

```
char* decomposer = “Beethoven”;
size_t decomp_length = strlen(decomposer);
char* composer = malloc(sizeof(char) * decomp_length);
strncpy(composer,decomposer,decomp_length);
```

10.10. When complete, the following program skeleton will search for the string “Windy” in the array place. If place contains “Windy” the program will display the message “Windy found.” Otherwise it will display “Windy not found.”

student’s note: the code has been edited in place instead of copied. because of, you know, laziness.

```
#include <iostream>
#include <string.h>
// include any other necessary header files
using namespace std;

int main()
{
  // Complete the program. It should search the array place
  // for the string "Windy" and display the message "Windy
  // found" if it finds the string. Otherwise, it should
  // display the message "Windy not found."


  char place[] = "The Windy City";
  char needle[] = "Windy";

  if (strnstr(place,needle,strlen(place))) {
    cout << "Windy found" << endl;
  } else {
    cout << "Windy not found" << endl;
  }
  return 0;
}
```

10.11. Write a short description of the following functions:

atoi - convert ASCII string to integer
atol - convert ASCII string to long or long long integer
atof - convert ASCII string to double
itoa - convert an integer to a string

10.12. Write a statement that will convert the string “10” to an integer and store the result int he variable num.

```
int num = atoi(“10”);
```

10.13. Write a statement that will convert the string “100000” to a long and store the result in the variable num.

```
long long int num = atol(“100000”);
```

10.14. Write a statement that will convert the string “7.2389” to a double and store the result in the variable num.

```
double num = atof(“7.2389”);
```

10.15. Write a statement that will convert the integer 127 to a string, stored in base-10 notation in the array value.

```
char* value = malloc(sizeof(char) * 4);
value = itoa(127,value,10);
```

10.16. What is the output of the following program?

```
#include <iostream>
using namespace std;

// Function Prototype
void mess(char []);

int main() {
      char stuff[] = "Tom Talbert Tried Trains";
      cout << stuff << endl;
      mess(stuff);
      cout << stuff << endl;
      return 0;
}

// Definition of function mess
void mess(char str[])
{
      int step = 0;
      while (str[step] != '\0')
      {
            if (str[step] == 'T')
                  str[step] = 'D';
            step++;
      }
}
```

The program outputs:

```
Tom Talbert Tried Trains
Dom Dalbert Dried Drains
```