Write blocks of code to perform the functions used in the following main program. Your blocks and title lines must match the function called in the main function. Each block should be a short function of only a few lines. [65 pts]

```cpp
int main() {
    string s = "HELLO", t = "GoodBye!";
    // (a) Tests whether a string omits the letter E, return true or false
    cout << hasNoE(s) << endl; //prints: 0
    // (b) Return the last character
    cout << last(t) << endl; //prints:
    // (c) Insert a space (" ") to a specified position of variable t
    insertSpace(t, 4); cout << t << endl; //prints: Good Bye!
    // (d) Returns parameter string with first letter removed
    cout << removeFirst(s) << endl; //prints: ELLO
    // (e) Returns the longest element in an array of strings
    string x[6] = { "This", "is", "an", "easy", "question", "" } ;
    cout << longestString(x, 6) << endl; // prints: question
    return 0;
}
```

**Answer:**

(1)

```cpp
bool hasNoE(string s) {
    //user function find, if substring “E” does not
    //exist in string s, it returns index -1
    return s.find("E") == -1;
    //or use a loop to check every letter as following
    //for (int i = 0; i < s.size(); i++)
    //  if (s[i] == ‘E’) return false;
    return true;
}
```

(2)

```cpp
char last(string s) {
    //last index of string s, is the length – 1.
    return s[s.size() - 1];
}
```

(3)

```cpp
void insertSpace(string &s, int i ){
    //parameter s is passed by reference
    s.insert(i, " ");
}
```

(4)

```cpp
string removeFirst(string s) {
    //returns the substring from the second character
    //to the end of the string.
    return s.substr(1);
}
```

(5)

```cpp
string longestString(string a[], int cap) {
    //assume the first string with index 0 in array a is
    //the longest
    int index = 0;
    //going through all remaining string index i, first
    //current string i’s length
    //is longer than assumed string index’s length,
    //then set i to be the index
    for (int i = 1; i < cap; i++)
        if (a[i].size() > a[index].size()) index = i;
    return index;
}
```
int main(int argc, char* argv[]) {
    int x[5] = {2,7,1,8,2};
    cout << x[3/2] * x[2] << endl; //line a
    string words[3] = {“An”, “easy”, “question”};
    for (int i = 0; i < 2; i++) cout << words[i]; cout << endl; //line b
    cout << words[0][0] << endl; //line c
    string s = argv[1];
    cout << s << endl; //line d
    cout << s.substr(1,2) << endl; //line e
    cout << argc << endl; //line f
    return 0;
}

What is the output at line a: 7
    //3/2 evaluates to 1 \rightarrow x[1] \times x[2] \rightarrow 7 \times 1 \rightarrow 7
What is the output at line b: An easy
    //i = 0, prints words[0] \rightarrow An, i increment to 1, prints words[1] \rightarrow easy
    //cout << endl; doesn’t belong to the body of the loop
What is the output at line c: A
    //words[0] is An , [0] is the first character
What is the output at line d: abc
    //argv[1], second argument of the command line
What is the output at line e: bc
    //sub string of s, starting at index 1, 2 characters
What is the output at line f: 3
    //there’re 3 arguments of the command line (each one separated by a space)