

Name:										
EmpID:										

1 (30 points) Answer the following questions.

- (1) Given `int arr[] = {-1, 6, 5, 3}`, what is `arr[2]`?

Answer: `arr[2]` is 5.

- (2) Declare function **foo** whose first formal parameter is a double variable named **d** (pass by reference) and second formal parameter is an int variable named **i** (pass by value) and there is no return type. You just need to write the function header, no implementation is needed.

Answer: `void foo(double& d, int i)`

- (3) What does `rand() % 3 + 2` produce?

Answer: `rand() % 3` generates a random integer 0, 1, or 2, so `rand() % 3 + 2` generates a random integer 2, 3, or 4.

- (4) Given `string greeting = "Wonderful"`; What is the value for `greeting.substr(2, 4)`?

Answer: `nder`

- (5) Suppose a runnable file is `prog`, write command to run it and redirect input from `data.txt`.

Answer: `prog < data.txt`

- (6) What is the value of `2 + 1 / 2 * 6.0`?

Answer: 2

- (7) Suppose `n` is 234, what the value of `digit` after the following two statements?

```
n /= 10;
digit = n % 10;
```

Answer: 3

(8) What is the output of the following code?

```
1 #include <iostream>
2 using namespace std;
3
4 int main() {
5     int sum = 0;
6     for (int i = -6; i < 2; i += 2)
7         sum += i;
8
9     cout << sum << endl;
10    return 0;
11 }
```

Answer: -12

(9) What is output for the following code?

```
1     int a = 5;
2     int* p = &a;
3     *p += 3;
4     cout << a << endl;
```

Answer: 8

(10) What is the output for the following code?

```
1 #include <iostream>
2 using namespace std;
3
4 void foo(int& a, int& b);
5
6 int main() {
7     int i = 0;
8     int j = 3;
9
10    foo(i, j);
11
12    cout << "i = " << i << ", j = " << j << endl;
13    return 0;
14 }
15
16 void foo(int& a, int& b) {
17     if (a < b) {
18         a++;
19         b--;
20     }
21
22 }
```

Answer: i = 1, j = 2

2 (15 points) Answer the following questions.

- (1) Write code to calculate $a^b + \sqrt{3}c$, and put the result to variable `c`. Assume that `a`, `b`, and `c` are properly defined and initialized double number. Hints: you may use `pow` and `sqrt` function.

Answer: `c = pow(a, b) + sqrt(3 * c);`

- (2) Write code to

- enter a file name from console,
- declare and instantiate an `ifstream` object, call it **fin**, to open that file to read.
- NO need to check whether the file can be opened correctly or not. Also no need to write statements to read from the file or close it.

Answer:

```
1 cout << "Enter a file name: ";
2 string fileName;
3 cin >> fileName; //no space in fileName, so cin >> fileName; is ok.
4 ifstream fin(fileName);
```

- (3) What is the output of the following code?

```
1 #include <iostream>
2 using namespace std;
3
4 int main() {
5     for (int row = 1; row <= 5; row++) {
6         for (int col = 0; col < row; col++)
7             cout << "#";
8
9         cout << endl;
10    }
11    return 0;
12 }
```

Answer:

```
#
##
###
####
#####
```

3 (30 points) Programming exercises

- (1) Write code in main to read a string (may contain spaces) from console, find out and print out the number of letter 'a' in it. For example, suppose we enter "apple orange", then print out 2.

Here is a sample input/output:

```
Enter a string: apple orange
appearances of a: 2
```

Answer:

```
1 //count number of appearances of letter 'a' in it.
2 //Sample input/output:
3 //Enter a string: apple orange
4 //appearances of a: 2
5
6 #include <iostream>
7 using namespace std;
8
9 int main() {
10     cout << "Enter a string: ";
11     string line;
12
13     getline(cin, line);
14
15     int count = 0;
16     for (int i = 0; i < line.size(); i++)
17         if (line[i] == 'a')
18             count++;
19
20     cout << "appearances of a: " << count << endl;
21
22     return 0;
23 }
```

- (2) Define a function, for a given array of integers and its size, find out whether all its elements are even or not. For example, call the above function on array with values 1, 2, 3, the return is false. Call the above function on array with values 2, 6, 8, 10, the return is true.

Answer:

```
1 #include <iostream>
2 using namespace std;
3
4 bool all_evens(int arr[], int size);
5
6 int main() {
7     int arr[] = {1, 2, 3};
8     int size = sizeof(arr) / sizeof(arr[0]);
9     cout << boolalpha << all_evens(arr, size) << endl; //false
10    //boolalpha print true for 1, false for 0.
11
12    int arr2[] = {2, 6, 8, 10};
13    int size2 = sizeof(arr2) / sizeof(arr2[0]);
14    cout << boolalpha << all_evens(arr2, size2) << endl; //even
15
16    return 0;
17 }
18
19 bool all_evens(int arr[], int size) {
20     for (int i = 0; i < size; i++) {
21         if (arr[i] % 2 != 0)
22             return false;
23     }
24
25     return true;
26 }
```

4 (15 points) Write codes

(12 points) Define a function named **range**, for a given array of integers and its size, return the difference of its largest and smallest elements.

Answer:

```
1 //Define a function to find out the range --
2 //difference between the largest and smallest elements --
3 //of an array.
4 //sample output:
5 //range of array is 11
6 #include <iostream>
7 using namespace std;
8
9 int range(int* arr, int size);
10
11 int main() {
12     int arr[] = {1, -2, 6, 9};
13     int size = sizeof(arr) / sizeof(arr[0]);
14
15     cout << "range of array is "
16           << range(arr, size) << endl;
17     return 0;
18 }
19
20 int range(int* arr, int size) {
21     int min = arr[0];
22     int max = arr[0];
23
24     for (int i = 1; i < size; i++)
25         if (arr[i] < min)
26             min = arr[i];
27         else if (arr[i] > max)
28             max = arr[i];
29
30     return max - min;
31 }
```

(3 points) In main function, declare and initialize array with values 1, -2, 6, 9. Call the above function to print out its return.

No need to include library, using namespace statement, or write main function header. Just write the code in main function.

5 (10 points) Write codes

Define a function, for a given string, return a string without digital letters in the original string.

For example, call the above function on “hello123, h1o2w a3r4e y5ou?”, the return is “hello, how are you?”.

You may use `int isdigit (int c);` to test whether character is a decimal digit or not.

Answer:

```
1 #include <iostream>
2 using namespace std;
3
4 string remove_digits(string str);
5
6 int main() {
7     cout << remove_digits("hello123, h1o2w a3r4e y5ou?") << endl;
8     return 0;
9 }
10
11 string remove_digits(string str) {
12     string result = "";
13     for (int i = 0; i < str.size(); i++) {
14         if (str[i] < '0' || str[i] > '9')
15             result += str[i];
16     }
17
18     return result;
19 }
```


Variable and Constant Definitions

Type	Name	Initial value
int	cans_per_pack	= 6;
const double	CAN_VOLUME	= 0.335;

Mathematical Operations

```
#include <cmath>
pow(x, y)    Raising to a power  $x^y$ 
sqrt(x)      Square root  $\sqrt{x}$ 
log10(x)     Decimal log  $\log_{10}(x)$ 
abs(x)       Absolute value  $|x|$ 
sin(x)       Sine, cosine, tangent of  $x$  ( $x$  in radians)
cos(x)
tan(x)
```

Selected Operators and Their Precedence

(See Appendix B for the complete list.)

[]	Array element access
++ -- !	Increment, decrement, Boolean <i>not</i>
* / %	Multiplication, division, remainder
+ -	Addition, subtraction
< <= > >=	Comparisons
= !=	Equal, not equal
&&	Boolean <i>and</i>
	Boolean <i>or</i>
=	Assignment

Loop Statements

```
while (balance < TARGET)
{
    year++;
    balance = balance * (1 + rate / 100);
}
```

Executed while condition is true

```
for (int i = 0; i < 10; i++)
{
    cout << i << endl;
}
```

```
do
{
    cout << "Enter a positive integer: ";
    cin >> input;
}
while (input <= 0);
```

Loop body executed at least once

Conditional Statement

```
if (floor >= 13)
{
    actual_floor = floor - 1;
}
else if (floor >= 0)
{
    actual_floor = floor;
}
else
{
    cout << "Floor negative" << endl;
}
```

Executed when condition is true

Second condition (optional)

Executed when all conditions are false (optional)

String Operations

```
#include <string>
string s = "Hello";
int n = s.length(); // 5
string t = s.substr(1, 3); // "ell"
string c = s.substr(2, 1); // "l"
char ch = s[2]; // 'l'
for (int i = 0; i < s.length(); i++)
{
    string c = s.substr(i, 1);
    or char ch = s[i];
    Process c or ch
}
```

Function Definitions

```
double cube_volume(double side_length)
{
    double vol = side_length * side_length * side_length;
    return vol;
}
```

Exits function and returns result.

```
void deposit(double& balance, double amount)
{
    balance = balance + amount;
}
```

Modifies supplied argument

Arrays

```
int numbers[5];
int squares[] = { 0, 1, 4, 9, 16 };
int magic_square[4][4] =
{
    { 16, 3, 2, 13 },
    { 5, 10, 11, 8 },
    { 9, 6, 7, 12 },
    { 4, 15, 14, 1 }
};

for (int i = 0; i < size; i++)
{
    Process numbers[i]
}
```

Vectors

```
#include <vector> Element type Initial values (C++ 11)
vector<int> values = { 0, 1, 4, 9, 16 };

vector<string> names; Initially empty

names.push_back("Ann"); Add elements to the end
names.push_back("Cindy"); // names.size() is now 2

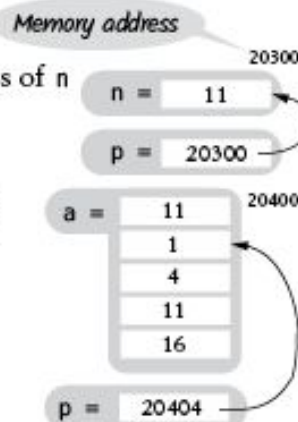
names.pop_back(); // Removes last element

names[0] = "Beth"; // Use [] for element access
```

Pointers

```
int n = 10;
int* p = &n; // p set to address of n
*p = 11; // n is now 11
```

```
int a[5] = { 0, 1, 4, 9, 16 };
p = a; // p points to start of a
*p = 11; // a[0] is now 11
p++; // p points to a[1]
p[2] = 11; // a[3] is now 11
```



Input and Output

```
#include <iostream>
cin >> x; // x can be int, double, string
cout << x;
```

```
while (cin >> x) { Process x }
if (cin.fail()) // Previous input failed
```

```
#include <fstream>
string filename = ...;
ifstream in(filename);
ofstream out("output.txt");
string line; getline(in, line);
char ch; in.get(ch);
```

```
void increment_print() {
    static int s_value = 0; //static duration
    s_value++;
    cout << s_value << '\n';
} //s_value is not destroyed, but goes out of scope

int main() {
    increment_print(); //1
    increment_print(); //2
}
```

Static Variables

```
class Item {
private:
    int m_id;
    static int s_id_counter;
public:
    Item() {
        m_id = s_id_counter++;
    }
    int get_id() const {
        return m_id;
    }
};

int Item::s_id_counter = 1;

int main() { //
    Item first;
    Item second;
    cout << first.get_id(); //1
    cout << second.get_id(); //2
}
```

Static Data Members

Range-based for Loop

```
An array, vector, or other container (C++ 11)
for (int v : values)
{
    cout << v << endl;
}
```

Output Manipulators

```
#include <iomanip>
```

```
endl Output new line
fixed Fixed format for floating-point
setprecision(n) Number of digits after decimal point
for fixed format
setw(n) Field width for the next item
left Left alignment (use for strings)
right Right alignment (default)
setfill(ch) Fill character (default: space)
```

Enumerations, Switch Statement

```
enum Color { RED, GREEN, BLUE };
Color my_color = RED;
```

```
switch (my_color) {
    case RED :
        cout << "red"; break;
    case GREEN:
        cout << "green"; break;
    case BLUE :
        cout << "blue"; break;
}
```

Class Definition

```
class BankAccount
{
public:
    BankAccount(double amount); Constructor declaration
    void deposit(double amount); Member function declaration
    double get_balance() const; Accessor member function
    ...
private: Data member
    double balance;
};

void BankAccount::deposit(double amount) Member function definition
{
    balance = balance + amount;
}
```

Inheritance

```
Derived class Base class
class CheckingAccount : public BankAccount
{
public:
    void deposit(double amount); Member function overrides base class
private:
    int transactions; Added data member in derived class
};

void CheckingAccount::deposit(double amount)
{
    BankAccount::deposit(amount); Calls base class member function
    transactions++;
}
```