ID: ....

Name: ....

Autumn 2019 – 1st In-Semester Examination

IT603 – Programming

Max. Marks: 20 Time: 90 mins

Instructions:

1. *This exam has 13 questions in total. All questions are compulsory.*
2. *All questions are to be answered in the question paper itself.*
3. *For objective type questions,* **circle the choice** *to mark your answer.*
4. *Write your ID on all pages of Question paper.*
5. *Question 1 through 8 carry 1 mark each.*
6. *The marks for remaining questions are mentioned alongside the questions.*
7. *For programs, you can assume headers are included and namespace std is used*

Q1. A C++ source code is dependent on a library named xyz.lib because it uses some functions from this library. When and by which tool/process these functions from the library are resolved for definitions?

a. Compiler during compilation
b. Linker during linking
c. Pre-processor during compilation
d. Operating System during runtime

Q2. Which of the following is the binary representation (LSB on right) of `0xA49C`

a. 1010 0100 1001 1100
b. 1100 1001 0100 1010
c. 1001 0100 1010 1100
d. 1010 1000 0101 0100

Q3. If you were to declare a variable named `bin` and initialize it to value 56 in binary format, how will do that in modern C++? The `bin` variable is supposed to have only positive values.

A. …………………………………………………………….

Q4. What is the output of following program?

```
void main()
{
      unsigned int r = 1;
      for (int i = 0; i < 4; ++i)
            std::cout << (r << i) << ' ' << (r >> i) << ' ';
}
```

a. 0 8 0 4 0 2 1 1
b. 1 1 2 0 4 0 8 0
c. 1 1 2 2 4 4 8 8
d. 1 1 1 1 1 1 1 1

Q5. Which of the following std::cout statement will display the text in next line exactly:

```
He said, "Here is the path: \\work\final.exe"
```

```
a. std::cout << He said, "Here is the path: \\work\\final.exe";
b. std::cout << "He said, "Here is the path: \\\\work\\final.exe";
c. std::cout << "He said, \"Here is the path: \\\\work\\final.exe\"";
d. std::cout << "He said, \\"Here is the path: \\work\final.exe\\"";
```

Q6. What is the output of following program?

```
void main()
{
      int a = 15, b = 6, c;
      c = ++a / b--;
      std::cout << ++c <<' '<< a++ <<' '<< --b;
}
```

a. 3.2 16 4
b. 3.2 17 5
c. 3 16 4
d. 3 17 4

Q7. In the following program, how many variables are created on stack within main()'s scope?

```
void main()
{
      int i = 10;
```

```
        int j = 20;
        int& k = i;
        int* m = &j;
}
```

a. 3
b. 4
c. 2
d. None of the above

Q8. Complete the following program using range based for loop so that it prints the values stored in the array to console.

```
void main()
{
        int arr[] = { 1,2,3,4,5 };

        for ( ………………………………………………………………)

                std::cout << ……………… << std::endl;

}
```

Q9. Complete the following program so that it doubles the value of an array element only if it is an even number. So the output when this program is run is

(2)

```
20 21 13 68 100 65 144
```

NOTE: Don't use `arr` variable or array notation in logic. Pointer variable should be used. Follow the comments.

```
void main()
{
   int arr[] = { 10, 21, 13, 34, 50, 65, 72 };

   int* p = ………………………………; // assign p to point to arr

   for (int i = 0; i < 7; ++i) {

      if (……………………………………) // check if p points to an even number

           …………………………; // if so, double it using p

      std::cout << …………………….. << ' '; // display the number using p

      ……………………………….; // let p point to next element of arr
   }
}
```

Q10. Observe the usage of pointers in the following program. Based on this usage, fill in the blank with most appropriate declaration of pointer variables especially with respect to `const`’ness.

(2)

```
void main()
{
      int i = 10;
      int j = 20;

      ……………………………………… cp1 = &i;   // cp1 is a pointer

      ……………………………………… cp2 = &j;   // cp2 is a pointer

      *cp2 -= 10;
      if (*cp1 == *cp2)
            cp1 = cp2;
}
```

Q11. Fill in the blanks in the following program such that the `dest` contains all strings of `src` but in reverse order.

(2)

```
void Q12()
{
  char src[5][5] = { "A a", "B b", "C c", "D d", "E,e" };
  char dest[5][5];

  for (int i = 0; i < 5; i++) {

    strcpy_s(…………………………, 5, ………………………………..);

    // strcpy_s - 1st param is destination, 3rd is source
  }
}
```

Q12. Complete the following program. It copies characters from `str` to `dest`, but while doing so it divides the `dest` in two parts (separated by a pipe ‘`|`’ character). The first part contains only vowel characters that occur in `str`, and second part contains non-vowel characters (including space, if any). In the example string “`An Advantage`” in code below, if the `dest` is printed to console at the end of this program, it will output something like:

```
AAaae|gtnvd n
```

Note: You can use int tolower(int) library function to convert any character to lowercase.

Important: Your code/algorithm should be O(N), i.e. It should at max be single loop. No loop within loop. Use pointers. Also, the order of characters does not matter. The only criteria is that left of pipe should be vowels and right of pipe rest of the characters.

```
void main()
{
  char str[100] = "An Advantage";
  char dest[100] = {};
```

(3)

```
  std::cout << dest << std::endl;
}
```

Q13. A company website need send user options back to server using bits. The first two bits (LSB side) are used for the credit card category – Signature, Gold, Platinum or Elite. The next bit (3rd LSB) is used for whether user opted for Credit Card. So, the first two bits make sense only if 3rd bit is set. Next bit (4th LSB) is used for Home Loan and 5th LSB for Car Loan.
On server side, the bits are stored in ‘`options`’ variable. But, the first (LSB side) byte of options variable is already used to store some other set of settings not related to this question. Therefore, you need to store the bits that are sent from client side in our case to the 2nd byte of `options` variable.
Complete the program below to do following tasks:

1. Copy the bits from `input` variable to `options` variable without affecting the bits already set in the first byte of `options` variable. The `input` variable is what is received by this server-side program as user options sent by client side.
2. Use `options` variable (and not the `input` variable) to then find out the settings set for a user. Eg. In this program the `input` variable is set to 21. So, the program should show following output:

```
Credit Card - Gold
Home Loan - NO
Car Loan – YES
```

(3)

Now complete the following program:

```
int main()
{
   typedef unsigned int uint;
   const int BYTE = 8;

   // assume the system has these bits set already in options.
   uint options = 219;

   // this is the input. you need to store this in options
   uint input = 21;   // no need to take user input
}
```