Q1. A C++ source code is statically 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 - correct

c. Pre-processor during compilation

d. Operating System during runtime

Mark: 2

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

a. 1010 0100 1001 1100 - correct

b. 1100 1001 0100 1010

c. 1001 0100 1010 1100

d. 1010 1000 0101 0100

Mark:2

Q3. Provide declaration of a function test() that –

- takes one parameter named 'arr' which is reference to a 2-dimensional char array of 5 rows and 6 columns
- returns a std::pair of an integer and a char.

Marks: 4

std::pair test(char(&arr)[5][6]);

Q4. Use range based for loop in program which uses the following array and updates it such that the even numbers get doubled, but no change is done to odd numbers.

int arr[] { 10, 21, 13, 34, 50, 65, 72 };

Then, use another range based for loop to print the arr array to console. The output when this program is run should look like:

20 21 13 68 100 65 144

Marks: 6

void Q4()

```
{
     int arr[]{ 10, 21, 13, 34, 50, 65, 72 };
    for (auto& cell : arr)
        cell = cell % 2 == 0 ? cell * 2 : cell;
    for (const auto& cell : arr)
        std::cout << cell << ' ';
}
```

Q5. For a 2-d matrix, the right most column should have sum of respective rows and bottom row should have sum of respective columns. eg:

20 -10 20 30

20 20 20 60

-10 -20 20 -10

30 -10 60 80

Write a program to do so. Use following hardcoded matrix 'm' as input. Use another matrix for preparing the result as above. You do not need to print anything to output.

```
int m[3][3] {{20,-10,20},
        {20,20,20},
        {-10,-20,20}};
```

Marks: 8

```
void Q5()
{
    int m[3][3]{  {20,-10,20},
            {20,20,20},
                {-10,-20,20} };
    int n[4][4];
    for (int i = 0; i < 3; i++)
        for (int j = 0; j < 3; j++)
            n[i][j] = m[i][j];
    for (int i = 0; i < 3; i++)
        n[i][3] = n[i][0] + n[i][1] + n[i][2];
        for (int i = 0; i < 4; i++)

                n[3][i] = n[0][i] + n[1][i] + n[2][i];

}
```

Q6. Write a program to create a 2-d array of 4 rows x 50 columns and copy following strings to it in order of their length, first being the string with least length. Your program should work correctly regardless of order of the strings in array below.

```
        char input[][50] {"Little bigger than medium",

                "Small",

                "This is longest string of all",

                "Medium string"};
```

Marks: 9

```
void Q6()

{

        char input[][50]{ "Little bigger than medium",

                "Small",

                "This is longest string of all",

                "Medium string" };

         const int rows = 4;

        char output[rows][50];

        std::pair lenIndex[rows]; // array of pairs, each pair stores length of string and its
        position/index in input array


        for (int i = 0; i < rows; i++)

        {

                lenIndex[i].first = strlen(input[i]);

                lenIndex[i].second = i;

        }

         // sort array of paris on length using bubble sort

        for (int i = 0; i < rows; i++)
```

```
	{
		for (int j = i; j < rows; j++)
		{
			if (lenIndex[j].first < lenIndex[i].first)
			{
				auto t = lenIndex[i];
				lenIndex[i] = lenIndex[j];
				lenIndex[j] = t;
			}
		}
	}
	// copy to output based on index in array of pair sorted on len

	for (int i = 0; i < rows; i++)
	{
		strcpy_s(output[i], strlen(input[lenIndex[i].second]) + 1,
	input[lenIndex[i].second]);
	}
}
```

Q7. Complete the following function for bitwise operations according to the comments provided. In you answer, write the whole function. You are not allowed to modify the code already written and it should be used in the portion that you write.

```
void bitwiseTest ()
{
	uint op{0b0111'0101'1010'0101}; // 30117
	cout << op << endl;
	uint bitVal{0};
	uint bcv{0};
	uint pos{0};
	// read the state of 3rd bit (2nd index) into bitVal
//<write code>
cout << bitVal << endl;
bitVal = pos = bcv = 0;
// set 10th bit
//<write code>
cout << op << endl;
bitVal = pos = bcv = 0;
 // reset 6th bit
//<write code>
cout << op << endl;
}
```

The output of this function on console is the following:

```
30117
1
30629
30597
```

Marks: 9

void Q7()

```
{
	unsigned int op{ 0b0111'0101'1010'0101 }; // 30117
	std::cout << op << std::endl;
	unsigned int bitVal{ 0 };
	unsigned int bcv{ 0 };
	unsigned int pos{ 0 };
	// read the state of 3rd bit (2nd index) into bitVal
	pos = 2;
	bcv = 1 << pos;
	bitVal = op & bcv;
	bitVal >>= pos;
	std::cout << bitVal << std::endl;
```

```
                    bitVal = pos = bcv = 0;

                    // set 10th bit
                    pos = 9;
                     bcv = 1 << pos;
                    op = op | bcv;
                    std::cout << op << std::endl;
                    bitVal = pos = bcv = 0;

                     // reset 6th bit
                    pos = 5;
                    bcv = 1 << pos;
                    op = op & ~bcv;
                    std::cout << op << std::endl;
}
void main()
{
          Q7();
}
```

1. Improve the signature of following function:

```
int X(int i, int j)
{
          return (i + j * i % j);
}
```

Ans:

```
int X(const int& i, const int& j)
{
          return (i + j * i % j);
}
```

2. These days vehicles in India have two kinds of number plates - HSRP (high security registration plate) and regular. HSRP has an unique ID (integer), which regular one doesn't have. We want to capture information of both number plates in a single structure. For regular plate, we want to store the word "regular" instead of ID (incase of HSRP). How will you do that?

Ans:

```
enum class RCType
{
  HSRP,
  REG,
}
struct RC
{
        RCType type;
        char* rcNum;
        union hsrp
        {
                int id;
                char* regular;
        }
};
```

3. Use the RC struct declared above to create few number plates on the Free Store by allocating memory properly and then deallocating at the end of the function / program.

```
void main()
{
        //consts
        const char* reg = "Regular";
        const int SIZE = 4;

        //input
        char rcValues[SIZE][] = {"GJ04EX2818", "GJ01AD0822", "GJ13CE8934", "GJ56EE5892"};
        int isHSRP[SIZE] = {0,1,1,0}; // corresponding rcValues is regular (0) or hrsp(1)

        // allocate and fill up data
        RC* rcs[SIZE];
        for(int i{0}; i < SIZE; i++)
        {
                rcs[i] = new RC;
                rcs[i]->type = isHSRP[i] == 0 ? RCType::REG : RCType::HSRP;
                auto l = strlen(rcValues[i])+1;
                rcs[i]->rcNum = new char[l];
                strcpy_s(rcs[i]->rcNum, l, rcValues[i]);
                if (rcs[i]->type == RCType::HSRP)
                {
                        rcs[i]->hsrp.id = rand()%100000;
                }
                else
                {
                        auto rl = strlen(reg)+1;
                        rcs[i]->hsrp.regular = new char[rl];
                        strcpy_s(rcs->hsrp.regular, rl, reg);
                }

        }

        // use the rcs ...

        // ...
```

```
        // done with rcs. delete memory allocated on FS
        for(int i{0}; i < SIZE; i++)
        {
                if (rcs[i]->type == RCType::REG)
                {
                        delete[] rcs[i]->hsrp.regular;
                }
                delete[] rcs[i]->rcNum;
                delete rcs[i];
        }
}
```

Q2. Write a program to count the number of words in a sentence. You do not need to take input from user. Use the following and print the result to console. The program should not count extra spaces eg. around word 'one'.

8

For simplicity, assume that you do not need to deal with sentences with leading/trailing spaces.

```
void main(){
char str[] = "The multi - word sentence could have more than  one  spaces between words";
        char c;
        int word = 1, check = 0,i = 0;

        while (str[i]) {
                c = str[i];
                if (isspace(c) && check == 0) {
                        word++;
                        check = 1;
                }
                else  {
                        check = 0;
                }
                i++;
        }
        cout << "Number of words : " << word << endl;
}
```

Q1. See the program below and find out data type and qualifier you think is most appropriate for the variables. Take hints from comments. When run, the program gives following output:

4

```
50000 with interest of 8.3 compounded quarterly, will become 96469.73 after 8
years.
```

```
void Q1()

{

        ………………………… r{8.3}; // interest in percent, fraction, constant value

        ………………………… p{}; // principal, large, only +ve amount, fraction

        ………………………… t{}; // duration in complete years, no fraction

        ………………………… n{4}; // compounded quarterly, constant value


        // example input:

        p = 50'000.00;

        t = 8;


        // a => future value

        auto a = p * pow((1 + (r/100.0f)/n), n*t);


        cout << p << " with interest of " << r << " compounded quarterly, will
become " << fixed << setprecision(2) << a <<" after " << t << " years.";

}
```

```
void main() {
        const float r{ 8.3 }; // interest in percent, fraction, constant value
        unsigned int p{}; // principal, large, only +ve amount, fraction
        int t{}; // duration in complete years, no fraction
        const int n{ 4 }; // compounded quarterly, constant value

        // example input:
        p = 50'000.00;
        t = 8;

        // a => future value
        auto a = p  pow((1 + (r / 100.0f) / n), n  t);

        cout << p << " with interest of " << r << " compounded quarterly, will become " << fixed << a << " after
" << t << " years.";
```

}

Q-20 Complete the two functions below

```
void ShowSearchResult(unorder_map<string, unsigned int> titletoCost, string& input)
{
        if (titletoCost.count(input) == 0) {
                cout << "Not Found";
        }
        else {
                cout << “Cost : ” << titletoCost[input] << “INR” << endl;
                }
        }
}
cin.get();
system("cls");
```

Q7. In C++ if a file is to be opened for output to, in append mode and the data to be written is binary how will you do that? Use example.bin as file name and myFile as variable name. Assume no path is required, just the file name would do. Also, no need to use include<>

ofstream myFile("example.bin", ios::binary, 'a');

Q8. Complete the Compact ByXOR function in the following program. It bitwise XORs-first, all the pages together

to first page-1.e. first cells of all pages are XORed and the result is stored in the first cell of first page

and so on. After that it takes the first page and compacts (l.e. XORs) all rows of the first page into first

row-l.e. all first cells of each row in the page are XORed and result stored in the first cell of first row and

so on. Then, in 3d loop, it XORs all cells of the first row to first cell of the row. Use concepts of page, row and column pointers.

```
const int R = 4, C = 3, P = 3;

int CompactByXOR(int (*p)[R][C])

//bitwise XOR the pages to compact them to 2d (in 1st page itself) // i.e. each cell of 1st page is bitwise-xored with same cell of the // other two pages and result is stored in the 1st page in that cell.

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

        for(int j=0; j<c; j++){

                *(*(*p+i)+j))^ = *(*(*(p+1)+i)+j) ^ *(*(*(p+2)+i)+j)
        }
}

// bitwise XOR the rows to compact them to 1d (in 1st row of 1st page itself)

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

        *(*(*p+0)+i)^ = *(*(*p+1) +i) ^ *(*(*p+2)+i) ^ *(*(*p+ 3) + i);

}

// bitwise XOR the cells of 1st row of 1st page to get the result in // first cell and return that

int result = 0;

result = *(*(*p+0)+0) ^ *(*(*p +0)+1) ^ *(*(*p+0) +2)
```

Q4. Which is correct declaration of function pointer fp to a function which takes constant pointer to an integer as parameter and returns integer?

a. int *fp (const int * intP)

b. int (*fp) (const int * intP) – correct

c. int *fp (int const * intP)

d. int (*fp) (int const* intP)

Q5. If you have in your program a function Twice that matches the signature of the

function pointer declared in Q4, then how would you let the function pointer of Q4 point to Twice function?

fp = (fp)Twise;

Q6. Imagine that you have Student structure. Fill in the code below for proper allocation and disposal of

free store memory

// create array of 10 Student pointers

Student *p[10];

// write for loop to create 10 Student nodes on free store and keep them in the array created above

```
for (int i = 0; i<10; i++)
{
        (p + i) = new Student ();
}
```

/* imagine that the Student objects are getting used by code represented by this comment */

// we are done with using the Student objects. Dispose all Student objects

```
for (int i{ 0 }; i < 10; i++)
  {
      delete[] p[i];
      p[i] = nullptr;
  }

  delete[] p;
```

14. Complete the function CpiCount below. The idea is to efficiently know how many students have a particular

cpi (like, 1.6 in the code below) and print the student's info of all such students. Use appropriate STL container.

```
void CpiCount()
{
        auto v = CreateSomeStudents();
        unordered_multimap<Student> cpis;

        for(auto s : v)
        {
                cpis.insert(s);
        }

        // create "empty" Student object but with the CPI we want to find Student checkCpl -("", 0, 8.6);

        int count = cpis.count(checkCpi); // write the function call
        cout << "Students with CPI" << checkCpi.cpi <<"=" << count << endl;

        auto p= cpis.equal_range(checkCpi);

        for (auto itr = p.first; itr != p.second; ++itr){
                PrintStudentinfo(*itr);
        }
}
```

Q12. Complete the following function which iterates over the vector v to create another sequence in which all female students are in the front of the queue and male students are placed after them efficiently. Iterate over this new sequence and print the info using PrintStudentinfo().

```
void FM()
{
        auto v= CreateSomeStudents();
        deque<student> dq;

        for (auto s: v) {

                if (s. gender = = female)
                        dq push_front(s);

                else
                        dq.push_back(s);
        }

        for (auto itr{dq.begin()}; itr! = dq.end(); itr++)
                PrintStudentInfo (*itr);
}
```

Q5. For the following definition of template function

```
template <typename T> T larger(T a, T b)
{
        return a>b? a:b;
}
```

how would you call this function passing it two string params a and b?

a) template<string> larger(string a, string b);

b) template<string>larger(a, b);

c) larger<string>(a, b); - correct

d) <string>larger(a, b);

Q6. For the template function defined in Q5, you specialize it for a type named Player. The prototype of this specialized template function would be

a) template<> Player larger(Player a, Player b); - correct

b) template<Player> Player larger(Player a, Player b);

c) template <typename T> Player larger(Player a, Player b);

d) template<class Player> Player larger(Player a, Player b);

Q7. How would you declare an iterator itr to a vector of elements where each element is a pair of string and integer?

a) iterator<vector<pair<string, int>::itr;

b) vector<pair<string, Int>>::iterator itr; - correct

c) auto itr;

d) auto::iterator<pair<string, Int> itr;

Q8. What is the output of following program?

```
void main()
{
        string s{"abcdef"};
        auto a = s.substr(1, 4);
        s[3] = 'z';
        a[0] = 'x';
        cout << a << ' ' << s << endl;
}
```

a) bcde abcdez

b) xcde abczef - correct

c) xcdz xbczef

d) abcd xbczef

09. Someone wrote the following program. You observed that it can be improved in terms of avoiding unnecessary

copies of the string objects. So, you commented the PrintSubString() function out:

```
/*
void PrintSubStringi(string fullString, int startindex, int count)
{
        if (fullString.length() >= startindex + count)

        {

                cout << fullString.substr(startindex, count);

        }

}

*/


void main()

{

        string s = "This is a long string";

        PrintSubString(s, 4, 10);

}
```

Rewrite the PringSubString() function that you as a programmer would write as an improvement to avoid copies of

strings.

void PrintsubString(string_view fullString, int start, int count)

{

if(fullString.length() >= Start + count)

{

cout << fullstring.substr(start, count);

}

}

```
void PrintsubString(string_view fullString, int start, int count)

{

        if(fullString.length() >= Start + count)

        {

                cout << fullstring.substr(start, count);

        }

}
```

For Q10 to 014, we'll use following partial program and you will be asked to complete / write the rest of the

program based on the questions.

```
enum Gender{

        female, male

};
```

```
struct Student{
        string name;
        long id;
        float cpi;
        Gender gender;
};

vold PrintStudentInfo(Student s)
{
        cout << s.id << '' << s.cpl << '' << s.name << '' << (s.gender == female ? "F" : "M") << endl;
}

vector<Student> CreateSomeStudents()
{

        vector<Student> v;
        v.push_back({"Shiraj Govil", 201612011, 8.6, male });
        v.push_back({"Ashita Patel", 201612021, 9.1, female));
        v.push_back({"Pusha Murthy", 201612041, 7.6, female });
        v.push_back({"Bibek Sen",201612001, 8.2, male });
        v.push_back({"Surit Saren",201612051, 8.6, male });
        return v;
}
```

Q10. Create a struct functor CompareByCpi which compares cpi of two students and returns true if student in first

parameter has cpi LESS than student passed in second parameter

```
Struct CompareByCpi {
        bool operator() (Const Student& s1, const Student& S2)
        {
                return s1.cpi < s2.cpi;
        }
```

};

Q11. Assuming struct CompareByCpi functor is available from Q10 above, complete the SortByCpi function below which

uses priority queue container and the CompareByCpi functor to print the list of students in vector v sorted by

their cpi. Use PrintStudentInfo() to print the student's info.

```
vold SortByCpi()
{
        auto v = CreateSomeStudents();

        Priority_queue<Student, vector<student>, CompareByCpi > sq;
        for(auto s : v)
        {
                sq.push(s);
        }
        while (!sq. empty())
        {
                PrintStudentInfo(sq.top());
                sq.poll();
        }
}
```

Q17. Complete the function below

```
// the authors in Book object could be comma separated list. So could be keywords.
// so we write a common function to extract list of sub-strings comma separated in given string

vector<string> SplitCommaSeparatedList (Const String& original)
{
        // you can assume there are no spaces around comma in original
        //i.e. original is like "a,b c,d" and not like "a, b c, d"
        vector<string> list;
        if (original.find (",") == String::npos) {//if there are no commas
                list.push_back(original); // add the only substr to list
        }
        else {// if there are one or more commas
                size_t startPos = 0, pos = 0;

                for(;;) {

                        auto pos= original.find(",", startPos); // find a comma

                        if(pos==String::npos){// check for end
                                // if no more commas, we need to copy the last author

                                if (startPos != original.length()) {
                                        list.push_back(original.Substr(startPos));
                                }
                                break; // break out of loop because we've reached the end
                        }
                        // add the extracted author to list
                        list.push_back(original.substr(startPos, pos));
                        startPos = pos + 1;
                }
        }
        return list;
}
```