Part 1 of 3 - LEVEL 1

Question 1 of 60
2.5 Points

Chose correct statement about below code:

```
struct MyStruct {
    int m1;
    int m2;
};

void myFunction(MyStruct* const myStruct) {
    myStruct ->m1 = 0;
    myStruct ->m2 = 0;
    // do other task here
}
```

A. This code can be compiled normally

B. This code cannot be compiled. Member of myStruct is private by default, so myFunction cannot access its member

C. This code can be compiled normally

D. This code cannot be compiled. myStruct pointer is const, so myFunction cannot access its member

YOUR ANSWER:

Your explanation: D

Part 1 of 3 - LEVEL 1

Question 2 of 60
2.5 Points

In below code, which type auto is deduced to?

```
const auto& x = 1;
```

A. int

B. int&

C. const int&

D. const int

YOUR ANSWER :

Your explanation: C

Part 1 of 3 - LEVEL 1

Question 3 of 60
2.5 Points

What is value of red, green, and yellow in below code?

```
enum class Color {red, green, yellow};

int main () {
    int red = 0;  int green = 0;  int yellow = 0;

    Color color = Color::green;
    switch (color) {
    case Color::red:
        red++;
    case Color::green:
        green++;
    case Color::yellow:
        yellow++;
    default:
        break;
    }
}
```

A. red = 0, green =1, yellow = 0

B. red = 1, green = 0, yellow = 1

C. red = 0, green =1, yellow = 1

D. red = 1, green = 1; yellow = 1

YOUR ANSWER :

Your explanation: C

Color color = Color::green

switch (color) sẽ nhảy vào case Color::green -> green++C
Ko có lệnh break -> yellow++

Part 1 of 3 - LEVEL 1

Question 4 of 60
2.5 Points

What is value of x and y in below code?

```
class MyClass {
    static int m1;
    int m2 = 0;
public:
    void add() {++m1; ++m2;}
    void get(int& arg1, int& arg2) {
        arg1 = m1; arg2 = m2;
    }
};

int MyClass::m1 = 0;

int main () {
    MyClass myClass1, myClass2;
    myClass1.add();
    myClass2.add();
    int x, y;
    myClass2.get(x, y);
}
```

A. x = 2, y = 1

B. x = 1, y = 1

C. x = 2, y = 2

D. x = 1, y = 2

YOUR ANSWER :

Your explanation: A
m1 là biến static của class nên sẽ cấp phát bộ nhớ tĩnh, biến độc lập qua các lời gọi hàm từ các đối tượng **myClass1, myClass2.**

Part 1 of 3 - LEVEL 1

Question 5 of 60
2.5 Points

Where are the place that p and memory that p point to (*p) allocated?

```
{
    int*p = new int(0);
}
```

A. p is allocated on stack, *p is allocated on heap

B. p and *p is allocated on heap

C. p and *p is allocated on stack

D. p is allocated on heap, *p is allocated on stack

YOUR ANSWER :

Your explanation: A

Giá trị của con trỏ (*p) nằm trên heap -> new- nam tren heap

Con trỏ p nằm trên stack

Part 1 of 3 - LEVEL 1

Question 6 of 60
2.5 Points

When destructor of myClass object is called?

```
class MyClass {
public:
    MyClass() {}
    ~MyClass() {}
};

int main() {
    MyClass myClass;
    // use myClass here…
}
```

A. Right after myClass definition

B. Will not be called

C. Before main() function exits

D. After main() function exits

YOUR ANSWER :

Your explanation: D

Destructor called when object **`MyClass`** destroyed

Part 1 of 3 - LEVEL 1

Question 7 of 60
2.5 Points

What is value of myVal in below code?

```
int main() {
    const char* myString = "hello";
    int myVal = myString[5];
}
```

A. 'o'

B. some unknown value

C. 0

D. Program may crash when accessing 5th element of myString

YOUR ANSWER :

Your explanation : C

Cuối **`const char*`** là \0

Part 1 of 3 - LEVEL 1

Question 8 of 60
2.5 Points

What is size of MyUnion instance?

```
union MyUnion {
    float a; // size of float is 4, alignment is 4
    char b; // size of char is 1, alignment is 1
```

A. 1

B. 4

C. 8

D. 6

YOUR ANSWER :

Your explanation: 4

Size of union= size của phần tử lớn nhất trong nó

Part 1 of 3 - LEVEL 1

Question 9 of 60
2.5 Points

What is value of y in below code?

```
#include <iostream>
int main() {
    int y;
    int x = 1;
    {
        int x = 2;
        y = x;
    }
    std::cout << "value of y is " << y << std::endl;
    return 0;
}
```

A. 1
B. 2
C. some unknown value
D. 0
YOUR ANSWER :

Your explanation: B

Part 1 of 3 - LEVEL 1

Question 10 of 60
2.5 Points

What is value of x in below code?

```
#include <iostream>

void release(int* p) {
    delete p;
}

int main() {
    int* x = new int(0);
    release(x);
    int y = *x;
    std::cout << y << std::endl;
}
```

A. 0

B. some unknown value

C. program behavior is undefined

D. program crashed when dereference x

YOUR ANSWER :

Your explanation: C

Truy cập con trỏ đã release, vẫn để nguyên vùng nhớ thuộc process này.

C++ calls undefined behaviour - you might be able to access the data, you might not.

Part 1 of 3 - LEVEL 1

Question 11 of 60
2.5 Points

What is value of *y in below code? (assume that y is 4 bytes arranged in little endian format)

```
char x[]= {0x11, 0x22, 0x33, 0x44};
int* y = (int*)(&x);
```

A. 0x44

B. 0x44332211

C. 0x11

D. 0x11223344

YOUR ANSWER :

Your explanation : B

Little endian: sx địa chỉ từ bé đến lớn- Lưu phần tử ở cuối lên đầu

Lưu đúng thứ tự: `0x11, 0x22, 0x33, 0x44`

Đọc đảo lại.

Part 1 of 3 - LEVEL 1

Question 12 of 60

2.5 Points

Chose correct statement about below code:

```
class MyClass {
public:
    virtual int getId() = 0;
};

int main () {
    MyClass myClass;
    std::cout << myClass.getId() << std::endl;
}
```

A. There is no problem with above code
B. This code cannot be compiled because MyClass is abstract class
C. This code can be compiled normally but runtime error will occur when getId() method is called
D. This code cannot be compiled. It is illegal to declare getId() function equal zero (getId() = 0)
YOUR ANSWER :

Your explanation: B

Có hàm pure vitual -> abtract class -> ko thể khởi tạo đối tượng **`MyClass myClass`**

Part 1 of 3 - LEVEL 1

Question 13 of 60
2.5 Points

Which constructor is called when constructing m2 object?

```
class MyClass {
    // implementation of MyClass here …
};

// use of MyClass
MyClass m1;
MyClass m2 = m1;
MyClass m2(m1);
```

A. Default constructor

B. Copy constructor

C. None of the above answer is correct

D. Move constructor

YOUR ANSWER :

Your explanation: B
if Copy assignment (m2=m1)

Part 1 of 3 - LEVEL 1

Question 14 of 60
2.5 Points

What is size of int in byte?

A. 4

B. 1

C. 8

D. sizeof(int)

YOUR ANSWER :

Your explanation: D
Tùy vào cấu trúc máy

(phụ thuộc vào compile or run time)

Part 1 sof 3 - LEVEL 1

Question 15 of 60
2.5 Points

Below code allocates 100 bytes of memory. How to release the allocated memory?

```
{
    void* p = malloc(100);
    // use p here…
}
```

A. free(p);

B. delete[] p;

C. No need to do anything

D. delete p;

YOUR ANSWER :

Your explanation: A

Part 1 of 3 - LEVEL 1

Question 16 of 60
2.5 Points

What is value of count after below code?

```
int main() {
    int count;
    for (count = 0; count < 10; ++count) {
        if(count == 5) continue;
    }
}
```

A. 0

B. 10

C. some unknown value

D. 5

YOUR ANSWER :

Your explanation: B

Part 1 of 3 - LEVEL 1

Question 17 of 60
2.5 Points

Why padding is needed in struct?

A. To improve overall application memory usage

B. To improve source code readability

C. To improve code size

D. To improve alignment

YOUR ANSWER :

Your explanation: D

Part 1 of 3 - LEVEL 1

Question 18 of 60
2.5 Points

What is value of myScore in below code?

```
#include <iostream>

void getScore(int& score) {
    score = 50;
}
//void getScore(int* score) {
//    *score = 50;
//}

int main() {
    int myScore;
    getScore(myScore);
    // getScore(&myScore);
    std::cout << "My score is " << myScore << std::endl;
    return 0;
}
```

A. 0

B. some unknown value

C. 50

D. -1

YOUR ANSWER :

Your explanation: C

Part 1 of 3 - LEVEL 1

Question 19 of 60
2.5 Points ++ A + A++

What is the result of x and y after below code?

```
int x = -1;
int y = -1;
bool increaseX() {return ++x;};
bool increaseY() {return ++y;};
int main () {
        if(increaseX() && increaseY()) {
        // do something here
}
        return 0;
}
```

A. x = -1, y = 0

B. x = 0, y = 0

C. x = 0, y = -1

D. x = -1, y = -1

3

YOUR ANSWER :

Your explanation : C

Part 1 of 3 - LEVEL 1

Question 20 of 60 search lại???
2.5 Points

What is linkage of x in below code?

```
int x;

int main() {
    // use x here…
}
```

A. External linkage

B. No linkage

C. Internal linkage

D. None of the above answer is correct

YOUR ANSWER :

Your explanation: A

Internal linkage refers to everything only **in scope of a translation unit**. (Local variable, Static variable, `const` global variable)

External linkage refers to things that exist beyond a particular translation unit. In other words, **accessible through the whole program**, which is the combination of all translation units (or object files) (Global variable, `extern const` global variable)

Part 2 of 3 - LEVEL 2

Question 21 of 60

1.5 Points

What is result of age in below code?

```
#include<iostream>

class Person {
public:
    virtual int getAge() {return 0;}
    virtual ~Person(){}
};

class Student : public Person {
public:
    int getAge() override {return 20;}
};

int main() {
    Person aStudent = Student();// Chuyển Student thanh Person ->gọi hàm
//copy constructor của Person
    int age = aStudent.getAge();
    return 0;
}
```

A. 0

B. Runtime error will occur

C. 20

D. Some unknown value

YOUR ANSWER :

Your explanation: A

```
Person *aStudent = new Student();
int age =  aStudent -> getAge();
```

Part 2 of 3 - LEVEL 2

Question 22 of 60
1.5 Points

What is value of c in below code?

```
#define MAX(a, b) ((a) > (b) ? (a) : (b))

int main() {
    int a = 1;
    int b = 2;
    int c = MAX(++a, ++b);
//(++a)>(++b) ? (++a) : (++b)
//2>3 ? (++a) : (++b)
//4
    return 0;
}
```

A. 2

B. 3

C. 4

D. 1

YOUR ANSWER :

Your explanation: C

Part 2 of 3 - LEVEL 2

Question 23 of 60 ??? khác nhau <> và “”
1.5 Points

Where does gcc preprocessor find included header <iostream> in following code?

```
// main.cpp
#include <iostream>
Include ”iostream”

int main() {
    std::cout << “hello” << std::endl;
    return 0;
}
```

A. Preprocessor looks for <iostream> in current directory, then preconfigured list of directory

B. Preprocessor only looks for <iostream> in current directory

C. Preprocessor looks for <iostream> in preconfigured list of directories, then current directory

D. Preprocessor only looks for <iostream> in preconfigured list of directories

YOUR ANSWER :

Your explanation: C

Part 2 of 3 - LEVEL 2

Question 24 of 60
1.5 Points

Which constructor is called when constructing myClass object?

```
class MyClass {
public:
    MyClass();
    MyClass(int);
    MyClass(const MyClass& other);
    MyClass(MyClass&& other);
    ~MyClass() {}
};

// use of MyClass
MyClass origin;
MyClass myClass(std::move(origin));
```

A. MyClass() is called

B. MyClass(MyClass&& other) is called

C. MyClass(const MyClass& other) is called

D. MyClass(int) is called

YOUR ANSWER :

Your explanation: B

```
std::move --- r-value &&
```

Part 2 of 3 - LEVEL 2

Question 25 of 60 lambda???
1.5 Points

What is result of area in below program?

```
#include <iostream>

void myFunction(int& s) {
    [&](int s) { s *= s;}(s);
}

int main() {
    int area = 10;
    myFunction(area);
    std::cout << area << std::endl;
    return 0;
}
```

A. None of above answer is correct

B. 100

C. 10

D. 1000

YOUR ANSWER :

Your explanation: C

```
#include <iostream>


void myFunction(int& s) {
   int c = 3;
   [](int &s) { s *= s;c++;}(s);
   std::cout<<c;
}


int main() {
   int area = 10;
   myFunction(area);
   std::cout << area << std::endl;
   return 0;
}
```

Lambda:

What is a lambda function?

The C++ concept of a lambda function originates in the lambda calculus and functional programming. A lambda is an unnamed function that is useful (in actual programming, not theory) for short snippets of code that are impossible to reuse and are not worth naming.

In C++ a lambda function is defined like this

```
[]() { } // barebone lambda
```

or in all its glory

```
[]() mutable -> T { } // T is the return type, still lacking throw()
```

`[]` is the capture list, `()` the argument list and `{}` the function body.

The capture list

The capture list defines what from the outside of the lambda should be available inside the function body and how. It can be either:

1. a value: [x]
2. a reference [&x]
3. any variable currently in scope by reference [&]
4. same as 3, but by value [=]

Part 2 of 3 - LEVEL 2

Question 26 of 60 ???
1.5 Points

Below code implementing copying two instances of MyClass. Chose correct statement about this implementation:

```
class MyClass {
public:
    MyClass() : mData(new int[10]) {}
    ~MyClass() {delete[] mData;}
    MyClass& operator=(const MyClass& other) {
        mData = other.getData();
        return *this;
    }
    int* getData() const {return mData;}
private:
    int* mData;
};
int main() {
    // use MyClass now
    MyClass other;
    MyClass myClass;
    myClass = other;
    std::cout << "class internal data" << myClass.getData() << std::endl;
    return 0;
}
```

A. This code has problem, double free error may occur because two instances of MyClass manage the same dynamic memory block

B. This code has problem, calling myClass.getData() may access invalid memory area and application will be terminated at this point

C. This code has no problem

D. This code has problem, memory leak may occur because pointer to dynamically allocated data is cloned between two instances of MyClass

YOUR ANSWER :

Your explanation: A/D

Part 2 of 3 - LEVEL 2

Question 27 of 60
1.5 Points

Chose correct statement about below code:

```
class Base {
public:
    virtual int getId() {return 0;}
    virtual ~Base() {}
};

class Derived : public Base {
public:
    virtual int getId(int) override {return 10;}
};
```

A. This code can be compiled normally

B. This code cannot be compiled, getId() function in Derived class declared override but no function in other classes override this function

C. This code cannot be compiled, it is illegal to declare getId() function in Derived class as both virtual and override

D. This code cannot be compiled, getId() function in Derived class declares override but it does not really override any function in base class

YOUR ANSWER :

Your explanation: B

Part 2 of 3 - LEVEL 2

Question 28 of 60
1.5 Points

In the following code, which template or function is taken after calling myFunc(&arg)?

```
// Template 1
template<typename T>
void myFunc(T arg);

// Template 2
template<typename T>
void myFunc(T* arg);

// Function
void myFunc(char* arg);

//use of myFunc here
int arg = 0;
myFunc(&arg);
```

A. `None of above is correct`

B. `void myFunc(char*)`

C. `void myFunc(T t);`

D. `void myFunc(T* t);`

YOUR ANSWER :

Your explanation: D

Part 2 of 3 - LEVEL 2

Question 29 of 60
1.5 Points

Given a source code with two files: main.cpp, other.cpp. What is the order of compilation between those two files?

A. There is no restriction in compilation order of the two files

B. Compilation of main.cpp and other.cpp are interleaving (if main.cpp uses some functions from other.cpp, compilation of main.cpp must wait for compilation of other.cpp before continuing)

C. Compile main.cpp then other.cpp

D. Compile other.cpp then main.cpp

YOUR ANSWER :

Your explanation: A

Part 2 of 3 - LEVEL 2

Question 30 of 60
1.5 Points

What is the value of x in below code?

```
#include<iostream>
struct MyStruct {
    int m1; int m2;
};

MyStruct& createX() {
    MyStruct me {-1, -1};
    return me;
}

int main (void){
    // use createX() to generate MyStruct instance;
    MyStruct& myStruct = createX();
    int x = myStruct.m1;
    return 0;
}
```

A. x is 0

B. x is -1

C. x is uninitialized

D. None of above answer is correct

YOUR ANSWER :

Your explanation: B

Khởi gán {}

Part 2 of 3 - LEVEL 2

Question 31 of 60
1.5 Points

What is result of x after below code?

```
class Base {
public:
    int getValue() const {return 0;}
    virtual ~Base(){}
};

class Derived : public Base {
public:
    int getValue() const {return 1;}
};
// use of Derived:
const Base& bRef = Derived();
int x = bRef.getValue();
```

A. Some unknown value

B. 1

C. 0

D. This code cannot be compiled because it is illegal to define getValue() function in both Base and Derived class

YOUR ANSWER :

Your explanation: C

Part 2 of 3 - LEVEL 2

Question 32 of 60
1.5 Points

Which constructors of MyMemberClass are called when constructing myClass object?

```
class MyMemberClass {
    int mData;
public:
    MyMemberClass() {}
    MyMemberClass(int data) : mData(data) {}
};

class MyClass {
    MyMemberClass mMember;
public:
    MyClass(int data) {
        // initialize mMember now…
        mMember = MyMemberClass(data);
    }
};

// use of MyClass
MyClass myClass(0);
```

A. MyMemberClass() is called

B. MyMemberClass(int data) is called, then MyMemberClass() is called

C. MyMemberClass(int data) is called

D. MyMemberClass() is called, then MyMemberClass(int data) is called

YOUR ANSWER :

Your explanation: D

Part 2 of 3 - LEVEL 2

Question 33 of 60
1.5 Points

Chose correct statement about below code

```
class Base {
public:
    virtual int getId() final {return 0;}
    virtual ~Base() {}
};

class Derived : public Base {
public:
    int getId () {return 100;}
};

int main() {
    Derived dObj;
}
```

A. This code can be compiled normally

B. This code cannot be compiled because getId() function is final in Base class

C. This code cannot be compiled because getId() function within Derived class does not have override specifier

D. This code cannot be compiled because constructor is not defined in Derived() class

YOUR ANSWER :

Your explanation: B

Part 2 of 3 - LEVEL 2

Question 34 of 60
1.5 Points

Function increaseX() needs to be thread-safety, that is, it can be called from multiple threads without suffering from data race. Which of following implementation satisfies that?

Implementation 1: x type is volatile int

```
void increaseX() {return ++x};
```

Implementation 2: x type is std::atomic

```
void increaseX() {return ++x;}
```

A. Neither Implementation 1 nor 2

B. Implementation 1

C. Implementation 2

D. Both Implementation 1 and 2

YOUR ANSWER :

Your explanation: C ???

Part 2 of 3 - LEVEL 2

Question 35 of 60
1.5 Points

Chose correct statement about below code:

```
#include <iostream>
#include <memory>

class MyClass {
public:
    int getValue() {return 0;}
};

int main () {
    MyClass *myClass = new MyClass();
    {
        auto sPtr = std::shared_ptr<MyClass>(myClass);
        std::cout << "my class data " << sPtr->getValue() << std::endl;
    }
    std::cout << "my class data " << myClass->getValue() << std::endl;
    return 0;
}
```

A. This code has problem, memory leak may occur because no one delete memory allocated for myClass

B. This code has no problem

C. This code has problem, crash may occur when calling getValue() function from myClass (myClass->getValue())

D. This code has problem, crash may occur when calling getValue() function from sPtr (sPtr->getValue())

YOUR ANSWER :

Your explanation: B

Part 2 of 3 - LEVEL 2

Question 36 of 60
1.5 Points

The below code cannot be compiled. Compiler error message is: "passing 'const MyClass' as 'this' argument discards qualifiers". What is the best way to fix this problem?

```
#include <iostream>
class MyClass {
public:
    int getData () {return m;}
private:
    int m;
};

void process(const MyClass& other) {
    std::cout << other.getData() << std::endl;
}

int main() {
    process(MyClass());
    return 0;
}
```

A. Add const in getData() member function of MyClass, that is:

```
int getData() const {return m;} // const is added
```

B. Use const_cast to cast away constness inside process() function, that is:

```
    void process(const MyClass& other) {
        std::cout << const_cast<MyClass&>(other).getData() << std::endl;
    }
```

C. Add mutable to declaration of m in MyClass, that is:

```
    mutable int m; // mutable is added
```

D. Remove const in process function parameter, that is:

```
    void process(MyClass& other) {…} // const is removed
```

YOUR ANSWER :

Your explanation: A

Part 2 of 3 - LEVEL 2

Question 37 of 60
1.5 Points

Chose correct statement about below code:

```
#include <iostream>
#include <memory>

class MyClass {
public:
    MyClass(int id) : mId(id) {}
    int getId(){
        return mId;
    }

private:
    int mId;
};

std::unique_ptr<MyClass> buildClass(int id) {
    return std::unique_ptr<MyClass>(new MyClass(id));
}

int main () {
    std::unique_ptr<MyClass> uPtr0 = buildClass(0);
    std::unique_ptr<MyClass> uPtr1 = buildClass(0);
    std::cout << "id0 " << uPtr0->getId() << ", id1 " << uPtr1->getId() <<
std::endl;
    return 0;
}
```

A. This code has problem, memory leak will occur because no one release memory allocated inside buildClass() function

B. This code has problem, double free error will occur because the allocated memory for MyClass instance is managed by both uPtr0 and uPtr1

C. This code has no problem

D. This code has problem, crash may occur when accessing getId() function from uPtr0 and uPtr1

YOUR ANSWER :

Your explanation: C

Part 2 of 3 - LEVEL 2

Question 38 of 60
1.5 Points

What is purpose of #ifndef #define #endif directive at the beginning and end of myheader.h?

```
//myheader.h
#ifndef MY_HEADER_H
#define MY_HEADER_H

// implementation of myheader.h here

#endif
```

A. To guide preprocessor how to interpret current header file

B. To guide preprocessor not to include myheader.h many times

C. None of above is correct

D. To guide preprocessor how to include myheader.h in source code file that uses this header

YOUR ANSWER :

Your explanation: B

Part 2 of 3 - LEVEL 2

Question 39 of 60
1.5 Points

What is value of x in below code?

```
#include <iostream>

class SimpleAdder {
public:
    int sum(int arg1, int arg2) {return arg1 + arg2;}
};

class SimpleAdderWithOffset : public SimpleAdder {
public:
    double sum(double arg1, double arg2) {return arg1 + arg2 + 1.0;}
};

int main () {
    SimpleAdderWithOffset myAdder;
    double x = myAdder.sum(1, 1);
    return 0;
}
```

A. 2

B. 3.0

C. 2.0

D. 3

YOUR ANSWER :

Your explanation: B/D???

Part 2 of 3 - LEVEL 2

Question 40 of 60
1.5 Points

What is the order of construction when creating Derived object dObj:

```
#include <iostream>

class A {};

class B {
public:
        B(int arg) {}
};

class Base {
public:
    virtual ~Base() {}
};

class Derived : public Base {
private:
    A a;
    B b;
public:
    Derived(int arg) : b(arg) {}
};

int main() {
    Derived dObj(0);
}
```

A. B(int), A(), Base(), Derived(int)

B. Derived(int), Base(), A(), B(int)

C. Base(), A(), B(int), Derived(int)

D. A(), B(int), Base(), Derived(int)

YOUR ANSWER :

Your explanation: D

Part 3 of 3 - LEVEL 3

Question 41 of 60
1.0 Points

What is value of rectArea and squareArea in below code?

```
#include<iostream>
class Shape {
protected:
    int area;
public:
    int getArea() { return area;}
    virtual ~Shape() {}
};

class Rectangle : virtual public Shape {
public:
    Rectangle() {area = 100;}
};

class Square : virtual public Shape {
public:
    Square() {area = 200;}
};

class Building : public Rectangle, public Square {
};

int main() {
    Building* bPtr = new Building();
    int rectArea = static_cast<Rectangle*>(bPtr)->getArea();
    int squareArea = static_cast<Square*>(bPtr)->getArea();
    delete bPtr;
    return 0;
}
```

A. rectArea = 200, squareArea = 100

B. rectArea = 200, squareArea = 200

C. rectArea = 100, squareArea = 100

D. rectArea = 100, squareArea = 200

YOUR ANSWER :

Your explanation: B

Virtual base

Part 3 of 3 - LEVEL 3

Question 42 of 60
1.0 Points

What is program behavior when data race occur?

A. Program crashed

B. Program runs normally

C. Program behavior is undefined

D. Program receives SIGSEGV signal. If signal handler is installed, after handling that signal, program continues to run normally

YOUR ANSWER :

Your explanation: D

Part 3 of 3 - LEVEL 3

Question 43 of 60
1.0 Points

What is result of x in below code?

```
class Shape {
public:
    virtual int getArea(int width = 0) const = 0;
    virtual ~Shape() {}
};

class Square : public Shape {
public:
    int getArea(int width = 1) const override {return width * width;}
};

int main() {
    const Shape& shape = Square();
    int x = shape.getArea();
}
```

A. Uninitialized value

B. 0

C. 2

D. 1

YOUR ANSWER :

Your explanation: B ???

Part 3 of 3 - LEVEL 3

Question 44 of 60
1.0 Points

Which function needs to implement to support calculating sum of many instances of MyComplex?

```
class MyComplex {
public:
    MyComplex(double real, double image) : mReal(real),  mImage(image){}
    MyComplex() : mReal(0), mImage(0) {}
    ~MyComplex() {}
private:
    double mReal;
    double mImage;
};

// use of MyComplex here
MyComplex  a1(1, 0);
MyComplex  a2(1, 1);
MyComplex  a3 = a1 + a2 + a1;  // a3 = (3, 1) = (1, 0) + (1, 1) + (1, 0)
```

A. void operator++(MyComplex& a1, const MyComplex& a2);

B. const MyComplex operator+(const MyComplex& a1, const MyComplex& a2);

C. void operator+=(MyComplex& a1, const MyComplex& a2);

D. const MyComplex& operator+(const MyComplex& a1, const MyComplex& a2);

YOUR ANSWER :

Your explanation: D ???

Part 3 of 3 - LEVEL 3

Question 45 of 60
1.0 Points

What is correct statement about below code?

```
// mymath.cpp
static double myCalculator(const double& arg) {return arg;}

// main.cpp
#include <iostream>
double myCalculator(const double& arg);
int main() {
    // use of myCalculator
    std::cout << myCalculator(1.0) << std::endl;
    return 0;
}
```

A. This code runs without problem

B. This code cannot be compiled because myCalculator function is not defined in main.cpp

C. This code cannot be linked because linker cannot find definition for myCalculator function

D. This code can be compiled and linked but run time error will occur when myCalculator function is called in main()

YOUR ANSWER :

Your explanation: C

Part 3 of 3 - LEVEL 3

Question 46 of 60
1.0 Points

Chose correct statement about allocating a variable on heap (X *x = new X();) and on stack (X x;)

A. Allocating on heap is slower but consuming less memory than on stack

B. Allocating on heap is faster and consuming less memory than on stack

C. Allocating on heap is faster but consuming more memory than on stack

D. Allocating on heap is slower and consuming more memory than on stack

YOUR ANSWER :

Your explanation: A

Part 3 of 3 - LEVEL 3

Question 47 of 60
1.0 Points

Compare below three implementations about runtime performance, which one is faster?

Implementation 1: square is normal function (suppose that square is not automatically inlined by compiler)

```
double square(double arg) {return arg * arg;}
double value = square(1.0);
```

Implementation 2: square is constexpr

```
constexpr double square(double arg) {return arg *arg;}
double value = square(1.0);
```

Implementation 3: square is inline (suppose that square is really inlined by compiler)

```
inline double square(double arg) {return arg * arg;}
double value = square(1.0);
```

(Note: “>” means faster than)

A. Implementation 3 > Implementation 1 > Implementation 2

B. Implementation 1 > Implementation 2 > Implementation 3

C. Implementation 2 > Implementation 3 > Implementation 1

D. Implementation 2 = Implementation 3 > Implementation 1

YOUR ANSWER :

Your explanation: D

(`constexpr` là từ khoá yêu cầu trình biên dịch tính toán phép tính hay biểu thức tại thời điểm biên dịch, trong khi đó `const` là công cụ giúp trình biên dịch nhắc nhở người dùng những biến nào không được thay đổi giá trị (nó chỉ nhắc nhở, có nghĩa là nếu ta muốn đổi cũng ok).)

Part 3 of 3 - LEVEL 3

Question 48 of 60
1.0 Points

What is the prefer address for allocating instance of MyStruct?

```
struct MyStruct {
    double x; // size: 8 bytes, alignment: 8 bytes
    char y; // size: 1 byte, alignment: 1 byte
}
```

A. At address that is multiples of 8

B. At any address

C. At address that is multiples of 4

D. At address that is multiples of 16

YOUR ANSWER :

Your explanation: A

Part 3 of 3 - LEVEL 3

Question 49 of 60
1.0 Points

Chose correct statement about shared_ptr and unique_ptr:

A. shared_ptr creation time is faster than unique_ptr and consumes less memory than unique_ptr

B. shared_ptr creation time is faster than unique_ptr but consumes more memory than unique_ptr

C. shared_ptr creation time is slower than unique_ptr but consumes less memory than unique_ptr

D. shared_ptr creation time is slower than unique_ptr and consumes more memory than unique_ptr

YOUR ANSWER :

Your explanation: D ???

Part 3 of 3 - LEVEL 3

Question 50 of 60
1.0 Points

What is value of x and y in below code?

```
#include <iostream>

template <typename T>
```

```
class MyCommon {
public:
    int increase() {return ++myBaseMember;}
private:
    static int myBaseMember;
};

template <typename T>
int MyCommon<T>::myBaseMember = 0;

class MyClassOne : public MyCommon<MyClassOne> {};
class MyClassTwo : public MyCommon<MyClassTwo> {};

int main() {
    MyClassOne myClassOne;
    MyClassTwo myClassTwo;
    int x = myClassOne.increase();
    int y = myClassTwo.increase();
    return 0;
}
```

A. x = 2 and y = 2

B. x = 1 and y = 2

C. x = 2 and y = 1

D. x = 1 and y = 1

YOUR ANSWER :

Your explanation: B

```
static int myBaseMember
```

Part 3 of 3 - LEVEL 3

Question 51 of 60
1.0 Points

What is result of x in below code?

```
#include <iostream>
#include <functional>
#include <math.h>

class MyMath {
```

```
public:
    void addSumFunc(std::function<double(double, double)> func) {mySum =
func;}
    double getSum(double arg1, double arg2) {return mySum(arg1, arg2);}
private:
    std::function<double(double, double)> mySum;
};

void addSum(MyMath& math) {
    double offset = 1.0;
    math.addSumFunc([&](double arg1, double arg2)->double{return arg1 + arg2
+ offset;});
}

int main() {
    MyMath m;
    addSum(m);
    double x = m.getSum(100, 100);
    return 0;
}
```

A. 201.0

B. 200.0

C. 100.0

D. None of above is correct

YOUR ANSWER :

Your explanation: ???

Part 3 of 3 - LEVEL 3

Question 52 of 60
1.0 Points

When declaration for g() in following template need to be found?

```
template<typename T>
class MyTemplate {
public:
    typename T::MyType a;
    typename T::MyType getA() {
        return g(a);
    }
};
```

A. Compiler finds declaration for g() at the time of template instantiation

B. Compiler finds declaration for g() at the time of template declaration

C. Compiler finds declaration for g() at the time of template definition

D. Compiler finds declaration for g() at the time getA() function is used

YOUR ANSWER :

Your explanation

Part 3 of 3 - LEVEL 3

Question 53 of 60
1.0 Points

When memory of allocated int object (10) is deallocated?

```
{
    std::weak_ptr<int> wptr;
    {
        auto sptr = std::shared_ptr<int>(new int(10));
        wptr = sptr;
    }
    if(auto tptr = wptr.lock()) {
        std::cout << "my managed pointer is " << tptr.get() << std::endl;
    }
}
```

A. When wptr is out of scope

B. When sptr is out of scope

C. Memory leak will occur

D. When program exits

YOUR ANSWER :

Your explanation: A

Part 3 of 3 - LEVEL 3

Question 54 of 60
1.0 Points

How many constructors of MyClass is called inside main() function?

```
#include <iostream>

class MyClass {
public:
    MyClass() {}
    ~MyClass() {}
};

MyClass getClass() {
    return MyClass();
}

int main() {
    // use of MyClass
    MyClass myClass = getClass();
}
```

A. 2 User-defined default constructor

B. 1 User-defined default constructor + 1 implicitly-defined copy assignment

C. 1 User-defined default constructor + 1 implicitly-defined copy constructor

D. 1 User-defined default constructor

YOUR ANSWER :

Your explanation: B

Part 3 of 3 - LEVEL 3

Question 55 of 60
1.0 Points

Does memory leak occur in below code?

```
#include <iostream>

class MyClass {
public:
    MyClass() {
        throw new std::runtime_error("error during constructor");
    }
};

int main() {
    try {
        MyClass* myClass = new MyClass();
    } catch (std::exception* e) {
        std::cout << e->what() << std::endl;
    }
}
```

A. Memory leak cannot occur because memory allocated for myClass object should be deallocated automatically

B. Memory leak cannot occur because memory allocated for exception object should be deallocated automatically

C. Memory leak can occur because no one deallocate memory of exception object

D. Memory leak can occur because no one deallocate memory of myClass object

YOUR ANSWER :

Your explanation

Part 3 of 3 - LEVEL 3

Question 56 of 60
1.0 Points

What is correct statement about below code?

```
#include <iostream>
```

```
class Person {
public:
    virtual void sayGoodbye() = 0;
    virtual ~Person() {sayGoodbye();}
};

class Student : public Person {
public:
    void sayGoodbye() override {std::cout << "bye bye" << std::endl;}
};

int main() {
    Student me;
    return 0;
}
```

A. Program can be compiled and linked normally but run time error will occur because of pure virtual function called

B. Program can be compiled, but the link step is failed because of pure virtual function called

C. Program cannot be compiled because of pure virtual function called

D. Program can be compiled, linked and run normally

YOUR ANSWER :

Your explanation: A

Part 3 of 3 - LEVEL 3

Question 57 of 60
1.0 Points

Chose correct statement about below code:

```
#include<iostream>
#include <memory>

void callSomeFunction() {throw std::runtime_error("error from outside");}

class MyClass {
public:
    MyClass() {
        mData = std::unique_ptr<int>(new int(0));
```

```
        callSomeFunction(); // may throw exception
    }
private:
    std::unique_ptr<int> mData;
};

int main () {
    try{
        MyClass myClass;
        // myClass.doSomething();
    } catch (std::exception& ) {}
    return 0;
}
```

A. This code has problem, memory leak may occur because on one release memory for exception object in the catch block

B. This code has problem, memory leak may occur because no one release memory for mData after exception is throw

C. This code has no problem

D. This code has problem, memory leak may occur because default destructor of MyClass does not release memory for mData

YOUR ANSWER :

Your explanation

Part 3 of 3 - LEVEL 3

Question 58 of 60
1.0 Points

Is memory leak occurs in following code?

```
#include<iostream>
#include <memory>
class Base {};
class Derived: public Base {
public:
    Derived() : mPtr(std::unique_ptr<int>(new int(0))) {}
    ~Derived() {}
private:
    std::unique_ptr<int> mPtr;
};
```

```
int main() {
    std::unique_ptr<Base>  bPtr = std::unique_ptr<Base>(new Derived());
    // bPtr->doSomething();
    return 0;
}
```

A. Memory leak cannot occur

B. Memory leak can occur because destructor of Derived class member will not be called

C. Memory leak can occur because no one deletes mPtr in destructor of Derived class

D. Memory leak can occur because no one deletes bPtr in main() function after it is used

YOUR ANSWER :

Your explanation: D

Part 3 of 3 - LEVEL 3

Question 59 of 60
1.0 Points

How many constructors and destructors of MyClass object are called inside the main() function?

```
#include <iostream>

class MyClass {
    int m;
public:
    MyClass(int arg) : m(arg) {std::cout << "hello " << m << std::endl;}
    ~MyClass() {std::cout << "goodbye " << m << std::endl;}
    void increase() {++m;}
};

void process(MyClass obj) {obj.increase();}

int main() {
    for(int i = 0; i < 10; ++i) {
        MyClass myClass(i);
        process(myClass);
    }
    return 0;
}
```

A. 10 Constructors + 10 Destructors

B. 30 Constructors + 30 Destructors

C. 1 Constructor + 1 Destructor

D. 20 Constructors + 20 Destructors

YOUR ANSWER :

Your explanation:

Part 3 of 3 - LEVEL 3

Question 60 of 60
1.0 Points

Chose correct statement about below code:

```
#include <iostream>
#include <thread>
#include <mutex>

std::string myString;
std::mutex myStringLock;

void setter() {
    std::lock_guard<std::mutex> guard(myStringLock);
    myString = "hello";
}

void getter() {
    std::cout << myString << std::endl;
}

int main () {
    std::thread thread1(setter);
    std::thread thread2(getter);
    thread1.join();
    thread2.join();
    return 0;
}
```

A. This code has problem because setter() function does not call mutex lock() before writing to myString

B. This code has no problem

C. This code has problem because setter () function does not call mutex unlock() after writing to myString

D. This code has problem because getter() function does not call mutex lock(), unlock() when reading myString

YOUR ANSWER :

Your explanation