1. Toán tử \_\_\_\_\_\_\_ sẽ tự động định vùng nhớ cho một đối tượng và trả về một \_\_\_\_\_\_\_
   1. **new pointer**
   2. pointer new
   3. new delete
   4. new free
2. Các hàm thành viên của một đối tượng có thể truy cập một con trỏ tự trỏ (seft pointer) tới đối tượng được gọi là con trỏ \_\_\_\_\_\_
   1. that
   2. \*
   3. **this**
   4. new
3. Từ khoá \_\_\_\_\_\_ định một biến hay một đối tượng không thể thay đổi được sau khi nó đã được khởi gán giá trị
   1. new
   2. **const**
   3. malloc
4. Một hàm thành viên có thể được khai báo là static nếu nó không truy cập các thành viên \_\_\_\_\_\_\_\_ của lớp
   1. static
   2. non-static
   3. **private**
5. Tìm và sửa lỗi

void ~Time ( int );

**// hàm hủy mặc định không được có giá trị trả về nên bỏ từ “void”**

* 1. void ~Time ();

**// hàm hủy mặc định không được có giá trị trả về nên bỏ từ “void”**

* 1. void ~( int );

**// hàm hủy mặc định không được có giá trị trả về nên bỏ từ “void”**

**// hàm hủy phải có tên mặc định là tên của Class**

* 1. void ~Time ( int t);

**// hàm hủy mặc định không được có giá trị trả về nên bỏ từ “void”**

**// hàm hủy mặc định không chứa tham số truyền vào**

* 1. void ~Time ( int &t);

**// hàm hủy mặc định không được có giá trị trả về nên bỏ từ “void”**

**// hàm hủy mặc định không chứa tham số truyền vào**

1. Tìm và sửa lỗi

class Time

{

public:

int hour = 0;

int minute = 0;

int second = 0;

};

* 1. class Time

{

private:

int hour = 0;

int minute = 0;

int second = 0;

};

* 1. class Time

{

public:

int hour;

int minute;

int second;

};

* 1. class Time

{

protected:

int hour = 0;

int minute = 0;

int second = 0;

};

* 1. class Time

{

public:

int hour = 0;

int minute;

int second;

};

1. Khai báo trong lớp Employee, tìm và sửa lỗi:

int Employee (const char \*, const char \*);

* 1. int Employee (char \*, char \*);

**// đặt tên Mothod không được trùng với tên của Class**

* 1. int Employee (const char , const char );

**// đặt tên Mothod không được trùng với tên của Class**

* 1. Employee (const char \*, const char \*);

**// đúng**

* 1. char Employee (const char \*, const char \*);

**// đặt tên Mothod không được trùng với tên của Class**

1. Chọn câu sai:
   1. Constructor có thể được khai báo const
   2. Destructor có thể được khai báo const
   3. Hàm friend của lớp có thể truy cập tất cả các thành viên private của lớp
   4. Một hàm hay lớp có thể là friend của một lớp khác
2. Chọn câu đúng:
   1. Mỗi đối tượng có thể truy cập đến chính nó thông qua con trỏ this
   2. Toán tử free được dùng để giải phóng vùng nhớ được cấp phát bởi new
   3. Thành viên static được dùng chung cho tất cả các thể hiện của lớp
   4. Các thành viên static của lớp chỉ tồn tại khi đối tượng của lớp tồn tại
3. Trong chương trình sau, hàm thiết lập nào được thực hiện

#include<iostream.h>

class XYZ {

private:

int value;

public:

XYZ() {value = 0;}

XYZ(int i = 0) {value = i;}

};

void main() { XYZ obj; }

* 1. Hàm thiết lập không tham số được thực hiện.
  2. Hàm thiết lập có 1 tham số được thực hiện.
  3. Trình biên dịch thông báo lỗi.
  4. Không có hàm thiết lập nào được gọi

1. Trong các hàm sau, hàm nào không phải là hàm constructor của lớp Phanso
   1. void Phanso(){};
   2. Phanso(const Phanso&x){};
   3. Phanso(){};
   4. Phanso(int x){};
2. Khi thực thi đoạn chương trình sau, kết quả sẽ là

class some {

int x;

public:

some() { x=10; }

~some() {

x--;

cout<<x<<endl;

}

};

void main()

{

some s;

s.~some();

}

* 1. 9 8
  2. 10 9
  3. 9
  4. 8 9