Задача. Дано: базовий клас **A**:

**class A**

**{ float x;**

**int y;**

**public:**

**A(): x(0.0), y(0) {}**

**float root() { return (sqrt(y);  }**

**protected:**

**void prn(void) { cout << x<< '  ' <<y; }**

**private:**

**char c;**

**};**

Визначити, як будуть успадковані елементи класу A у похідному класі B i D:

class B: public A {};

class D: protected B {};

Відповідь

Клас B успадковує функцію root(), яка має public тип доступу і функцію prn(void) яка має protected тип доступу.

Клас D успадковує фукції root() і prn(void), які мають protected тип доступу.

Задача 3

#include <iostream>

#define MAX\_COUNT 10

class Array

{

public:

int count;

Array();

Array(int count);

Array(int count, unsigned char value);

~Array();

unsigned char& operator[](int);

virtual Array Plus(Array & other);

protected:

unsigned char\* arr;

};

Array::Array()

{

this->count = 1;

arr = new unsigned char(count);

arr[0] = 0;

}

Array::Array(int count)

{

if (count > MAX\_COUNT)

{

std::cout << "Invalid count";

exit(0);

}

this->count = count;

arr = new unsigned char(count);

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

{

arr[i] = 0;

}

}

Array::Array(int count, unsigned char value)

{

if (count > MAX\_COUNT)

{

std::cout << "Invalid count";

exit(0);

}

this->count = count;

arr = new unsigned char(count);

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

{

arr[i] = value;

}

}

Array::~Array()

{

//delete[] arr;

}

unsigned char& Array::operator[](int index)

{

if (index >= count) {

std::cout << "Array index out of bound";

exit(0);

}

return arr[index];

}

Array Array::Plus(Array& other){

if (this->count != other.count)

{

std::cout << "The sizes do not match";

exit(0);

}

Array result(count);

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

{

result[i] = arr[i] + other[i];

std::cout << "r[" << i << "]: " << (int)result[i] << "\n";}

return result;}

class Decimal : public Array

{

public:

Decimal():Array() {};

Decimal(int count):Array(count) {};

Decimal(int count, unsigned char value):Array(count, value) {};

Array Plus(Array& other) override;

};

Array Decimal::Plus(Array& other)

{

if (this->count != other.count)

{

std::cout << "The sizes do not match";

exit(0);

}

Array result(count);

unsigned char carry = 0;

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

unsigned char sum = arr[i] + other[i] + carry;

result[i] = sum % 10;

carry = sum / 10;

std::cout << "r[" << i << "]: " << (int)result[i] << "carry: " << (int)carry << "\n";

}

if (carry != 0) {

std::cout << (int)carry << "Decimal addition produces overflow\n";

//exit(0);

}

return result;

}

class Hex : public Array

{

public:

Hex() :Array() {};

Hex(int count) :Array(count) {};

Hex(int count, unsigned char value) :Array(count, value) {};

Array Plus(Array& other) override;

};

Array Hex::Plus(Array& other)

{

if (this->count != other.count){

std::cout << "The sizes do not match";

exit(0);}

Array result(count);

unsigned char carry = 0;

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

unsigned char sum = arr[i] + other[i] + carry;

result[i] = sum % 16;

carry = sum / 16;

std::cout << "r[" << i << "]: " << (int)result[i] << "carry: " << (int)carry << "\n";

}

if (carry != 0) {std::cout << "Hex addition produces overflow\n"; exit(0);}

return result;}

int main()

{

Array a(3);

a[0] = 1;

a[1] = 2;

a[2] = 3;

Decimal b(3);

b[0] = 4;

b[1] = 5;

b[2] = 6;

Hex c(3);

c[0] = 0x7;

c[1] = 0x8;

c[2] = 0x9;

std::cout << "a+a\n";

a.Plus(a);

std::cout << "a+b\n";

a.Plus(b);

std::cout << "a+c\n";

a.Plus(c);

std::cout << "b+a\n";

b.Plus(a);

std::cout << "b+b\n";

b.Plus(b);

std::cout << "b+c\n";

b.Plus(c);

std::cout << "c+a\n";

c.Plus(a);

std::cout << "c+b\n";

c.Plus(b);

std::cout << "c+c\n";

c.Plus(c);}