ĐÁP ÁN – THANG ĐIỂM

Câu 1.

a. Hàm thuần ảo là gì? Lớp trừu tượng là gì? Cho ví dụ minh họa. (1đ)

Hàm thuần ảo (Phương thức ảo thuần tuý) có ý nghĩa cho việc tổ chức sơ đồ phân cấp các lớp, nó đóng vai trò chừa sẵn chỗ trống cho các lớp con điền vào với phiên bản phù hợp. Phương thức ảo thuần tuý là phương thức ảo không có nội dung, được khai báo với từ khóa virtual và được gán giá trị =0

Khi lớp có phương thức ảo thuần tuý, lớp trở thành lớp cơ sở trừu tượng. Lớp cơ sở trừu tượng không có đối tượng nào thuộc chính nó.

```
class Shape    //Abstract
{
   public :
   //Pure virtual Function
   virtual void draw() = 0;
}
```

Trong ví dụ trên, các hàm thành phần trong lớp Shape là phương thức ảo thuần tuý và lớp Shape là lớp cơ sở trừu tượng. Nó bảo đảm không thể tạo được đối tượng thuộc lớp Shape.

b. Trình bày các đặc điểm quan trọng của lập trình hướng đối tượng (1 điểm)

- Trừu tượng hóa – Abstraction Cách nhìn khái quát hóa về một tập các đối tượng có chung các đặc điểm được quan tâm (và bỏ qua những chi tiết không cần thiết).
- Đóng gói – Encapsulation Nhóm những gì có liên quan với nhau vào làm một, để sau này có thể dùng một cái tên để gọi đến. Vd: các hàm/ thủ tục đóng gói các câu lệnh, các đối tượng đóng gói dữ liệu của chúng và các thủ tục có liên quan.
- Thừa kế - Inheritance cho phép một lớp D có được các thuộc tính và thao tác của lớp C, như thể các thuộc tính và thao tác đó đã được định nghĩa tại lớp D. Cho phép cài đặt nhiều quan hệ giữa các đối tượng: Đặc biệt hóa – Tổng quát hóa
- Đa hình – Polymorphism Là cơ chế cho phép một tên thao tác hoặc thuộc tính có thể được định nghĩa tại nhiều lớp và có thể có nhiều cài đặt khác nhau tại mỗi lớp trong các lớp đó.

Câu 2.

Xây dựng lớp Đa thức bậc n (1đ) với các toán tử >> (0.5đ), << (0.5đ), + (1đ)
Sv có thể giải bằng dữ liệu mảng cấp phát động hoặc tĩnh đều được chấp nhận. Vì đề bài là đa thức bậc n nên thuộc tính của đa thức phải lưu trữ gồm danh sách nhiều hệ số. Các trường hợp chỉ có n, a, b, c là không đạt yêu cầu (vì chỉ dùng được cho đơn thức ax+b hoặc đa thức bậc 2: ax^2 + bx + c, …)

```
class DaThuc
{
	private:
		int mu;
		float *hs;
	public:
		friend istream & operator >>(istream &is,DaThuc &d);
		friend ostream & operator <<(ostream &os,DaThuc d);
		friend DaThuc operator +(DaThuc,DaThuc);

};

istream & operator >>(istream &is,DaThuc &d)
{
cout<<"\n Nhap so mu:";
do
{
is>>d.mu;
}
while(d.mu<=0);
d.hs=new float[d.mu+1];

for(int i=0;i<=d.mu;i++)
{
cout<<"\n X^"<<i<<"=";
is>>d.hs[i];
}
return is;
}

ostream & operator <<(ostream &os,DaThuc d)
{
for(int i=d.mu;i>1;i--)
{
if(d.hs[i]!=0)
os<<d.hs[i]<<"X^"<<i;
if(d.hs[i-1]>0)
os<<"+";
}
```

```
os<<d.hs[1]<<"X";
if(d.hs[0]>0)
os<<"+"<<d.hs[0];
else
os<<d.hs[0];

return os;
}
DaThuc operator +(DaThuc x,DaThuc y)
{
DaThuc kq;
int moc;
if(x.mu==y.mu)
{
kq.mu=x.mu;
for(int i=0;i<=kq.mu;i++)
kq.hs[i]=x.hs[i]+y.hs[i];
}
else

{ if(x.mu>y.mu)
{
kq.mu=x.mu;
moc= y.mu+1;
}
else
{
kq.mu=y.mu;
moc=x.mu+1;
}
for(int i=0;i<=moc;i++)
kq.hs[i]=x.hs[i]+y.hs[i];
for(i=moc;i<=kq.mu;i++)
if(x.mu>y.mu)
kq.hs[i]=x.hs[i];
else
kq.hs[i]=y.hs[i];
}
return kq;
}
```

Câu 3.

Có nhiều cách để thiết kế và viết chương trình cho câu 3. Sv có sử dụng kế thừa, đa hình và thực hiện 3 yêu cầu của đề thi được xem là đạt yêu cầu.

-Vẽ sơ đồ lớp đối tượng (1.5đ)

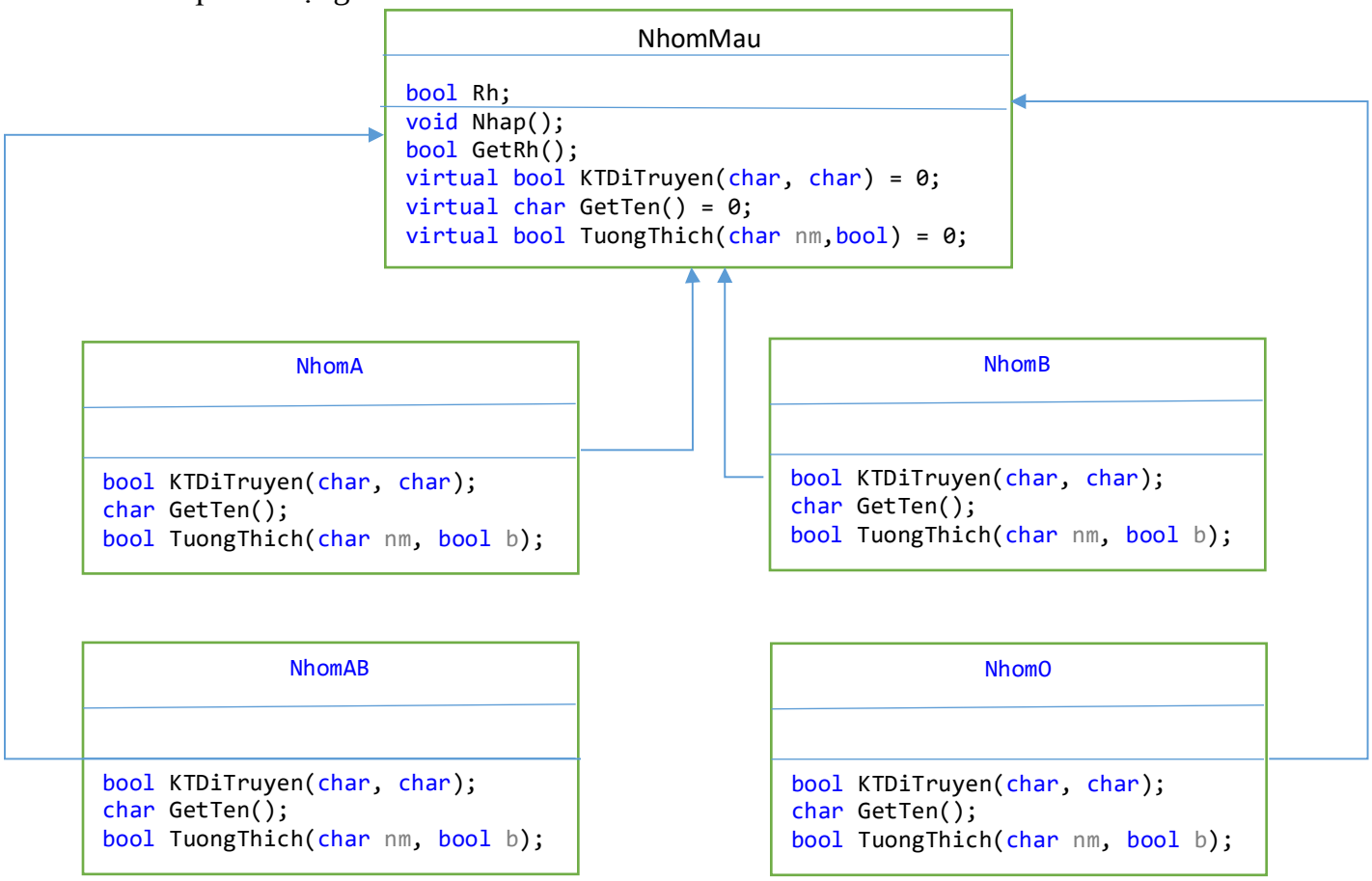
-Nhập danh sách nhóm máu (1.5đ)

-Kiểm tra quy luật di truyền (1đ)

-Danh sách người có thể cho máu người X (1đ)

Dưới đây là một cách giải cho câu 3:

Sơ đồ lớp đối tượng



Chương trình:

```
class NhomMau
{
protected:
	bool Rh;
public:
	NhomMau();
	~NhomMau();
	void Nhap();
	bool GetRh();
```

```
	virtual bool KTDiTruyen(char, char) = 0;
	virtual char GetTen() = 0;
	virtual bool TuongThich(char nm,bool) = 0;
};

NhomMau::NhomMau()
{
}


NhomMau::~NhomMau()
{
}

void NhomMau::Nhap()
{
	char t;
	cout << "Nhap Rhesus";
	cin >> t;
	if (t == '+')
		Rh = true;
	else
		Rh = false;
}

bool NhomMau::GetRh()
{
	return Rh;
}


class NhomA :
	public NhomMau
{
public:
	NhomA();
	~NhomA();
	bool KTDiTruyen(char, char);
	char GetTen();
	bool TuongThich(char nm, bool b);
};

NhomA::NhomA()
{
}


NhomA::~NhomA()
{
}
char NhomA::GetTen()
{
	return 'A';
}
bool NhomA::TuongThich(char nm, bool b)
{
	if (this->GetRh() == false)
		if(nm=='B'||nm=='C')
```

```
                        return true;
        if (this->GetRh() == true)
                if (b == true)
                        if(nm=='A'||nm=='C')
                                return true;
        return false;
}
bool NhomA::KTDiTruyen(char me, char con)
{
        switch (me) {
        case 'A': if (con == 'A' || con == 'O')
                                        return true;
                break;
        case 'B': if (con == 'A' || con == 'O' || con == 'B' || con == 'C')
                                        return true;
                break;
        case 'C': if (con == 'A' || con == 'B' || con == 'C')
                                        return true;
                break;
        case 'O':       if (con == 'A' || con == 'O')
                                        return true;
                break;
        }
        return false;
}

class NhomB :
        public NhomMau
{
public:
        NhomB();
        ~NhomB();
        bool KTDiTruyen(char, char);
        char GetTen();
        bool TuongThich(char nm, bool b);
};
NhomB::NhomB()
{
}


NhomB::~NhomB()
{
}
char NhomB::GetTen()
{
        return 'B';
}
bool NhomB::TuongThich(char nm, bool b)
{
        if (this->GetRh() == false)
                if (nm == 'B' || nm == 'C')
                        return true;
        if (this->GetRh() == true)
                        if (b == true)
                                if (nm == 'A' || nm == 'C')
                                        return true;
        return false;
```

```
}
bool NhomB::KTDiTruyen(char me, char con)
{
	switch (me) {
	case 'A': if (con == 'A' || con == 'O' || con == 'B' || con == 'C')
		return true;
		break;
	case 'B': if(con == 'B' || con == 'O')
		return true;
		break;
	case 'C': if (con == 'A' || con == 'B' || con == 'C')
		return true;
		break;
	case 'O':	if (con == 'B' || con == 'O')
		return true;
		break;
	}
	return false;
}

class NhomAB :
	public NhomMau
{
public:
	NhomAB();
	~NhomAB();
	bool KTDiTruyen(char, char);
	char GetTen();
	bool TuongThich(char nm, bool b);
};

NhomAB::NhomAB()
{
}


NhomAB::~NhomAB()
{
}
char NhomAB::GetTen()
{
	return 'C';
}
bool NhomAB::TuongThich(char nm, bool b)
{
	if (this->GetRh() == false)
		if (nm == 'C')
			return true;
	if (this->GetRh() == true)
		if (b == true)
			if (nm == 'C')
				return true;
	return false;
}
bool NhomAB::KTDiTruyen(char me, char con)
{
	switch (me) {
```

```
	case 'A': if (con == 'A' || con == 'B' || con == 'C')
		return true;
		break;
	case 'B': if (con == 'A' || con == 'B' || con == 'C')
		return true;
		break;
	case 'C': if (con == 'A' || con == 'B' || con == 'C')
		return true;
		break;
	case 'O':	if (con == 'A' || con == 'B')
		return true;
		break;
	}
	return false;
}
class NhomO :
	public NhomMau
{
public:
	NhomO();
	~NhomO();
	bool KTDiTruyen(char, char);
	char GetTen();
	bool TuongThich(char nm, bool b);
};

NhomO::NhomO()
{
}


NhomO::~NhomO()
{
}

bool NhomO::KTDiTruyen(char me, char con)
{
	switch (me) {
	case 'A': if (con == 'A' || con == 'O' )
		return true;
		break;
	case 'B': if (con == 'B' || con == 'O' )
		return true;
		break;
	case 'C': if (con == 'A' || con == 'B' )
		return true;
		break;
	case 'O':	if (con == 'O' )
		return true;
		break;
	}
	return false;
}

char NhomO::GetTen()
{
	return 'O';
}
```

```
bool NhomO::TuongThich(char nm, bool b)
{
	if (this->GetRh() == false)
		return true;
	if (b == true)
		return true;
	return false;
}
int main()
{
	//cau 1
	int n,chon;
	NhomMau *DS[50];
	cout << "Nhap so nguoi";
	cin >> n;
	for (int i = 0; i < n; i++)
	{
		cout << "Hay chon 1 cho nguoi nhom mau O";
		cout << "Hay chon 2 cho nguoi nhom mau A";
		cout << "Hay chon 3 cho nguoi nhom mau B";
		cout << "Hay chon 4 cho nguoi nhom mau AB";
		cin >> chon;
		switch (chon) {
			case 1: DS[i] = new NhomO();
				break;
			case 2: DS[i] = new NhomA();
				break;
			case 3: DS[i] = new NhomB();
				break;
			case 4: DS[i] = new NhomAB();
				break;
		}
		DS[i]->Nhap();
	}
	//cau 2
	int cha, me, con;
	cout << "Hay nhap theo thu tu cha, me, con";
	cin >> cha >> me >> con;
	bool KQ = DS[cha]->KTDiTruyen(DS[me]->GetTen(), DS[con]->GetTen());

	//cau 3
	int x;
	cout << "Nhap x";
	cin >> x;
	for (int i = 0; i < n; i++)
		if ((i != x) && (DS[x]->TuongThich(DS[i]->GetTen(), DS[i]->GetRh())))
			cout << "\t"<<i;
    return 0;
}
```