Писмен изпит по ООП/*04.06.2010*

Вариант А

Задача 1.

I. Да се реализира клас **Patient**, който съдържа информация за пациент: име, адрес и ЕГН. За този клас да се реализират:
   1) голяма четворка и конструктор с параметри;
   2) подходящи селектори и мутатори;
   3) функции за въвеждане и извеждане.

II. Производен клас **SickPatient**, която съдържа допълнителна информация за болен пациент: диагноза и ден на заболяване от началото на годината (число от 1 до 366). Да се допълнят функциите от а) - в) за производния клас.

III. Да се реализира клас **GP**, който съдържа два списъка: от пациенти и от болни пациенти. За този клас да се реализират:
   1) голяма четворка;
   2) функции за въвеждане и извеждане;
   3) функция за добавяне на нов пациент;
   4) функция за прехвърляне на здрав пациент в списъка от болните пациенти;
   5) функция за връщане на болен пациент в списъка на здравите пациенти;
   6) функция за отпечатване на всички пациенти с дадена диагноза;
   7) функция за сортиране на всички пациенти по възраст;
   8) функция, която намира сумата от всички болнични дни на всички болни пациенти до момента (днешният ден се задава като параметър).

Задача 2.

I. Да се реализира абстрактен клас **Task**, описващ задача. Всяка задача да поддържа следните операции:
   1) bool finished () – приключила ли е задачата;
   2) void doWork () – извършване на работа по задачата;
   3) bool canStart () – може ли да започне работа по задачата в текущия момент;
   4) void print () – отпечатване на информация за задачата.

II. Да се напишат две реализации на **Task** – **StepTask** и **DependantTask**.
   1) Клас **StepTask** реализира задача, която приключва след определен брой извършвания на работа по нея (брой извиквания на метода doWork). Необходимият броя стъпки се задава при конструиране на задачата. При отпечатване се предоставя информация за името и номера на задачата.
   2) Клас **DependantTask** реализира задача, която приключва за една единствена стъпка (извикване на метода doWork), но не може да започне, преди друга определена задача да е приключила. Блокиращата задача се подава при конструиране на зависимата задача. При отпечатване се предоставя информация за името и номера на задачата и информацията за блокиращата задача. Обърнете внимание, че цикли са невъзможни!

III. Да се реализира функция void completeAll (Task *[], int n), която по масив от задачи довежда всички до приключило състояние. Функцията итеративно изпълнява стъпки от всички задачи, които могат да бъдат започнати в дадения момент. Процесът продължава до тогава, докато броят на задачите, които могат да започнат, стане 0. Ако в този момент има задачи, които не са приключили да се отпечата информация за тях.

Писмен изпит по ООП/*04.06.2010*

Вариант Б

Задача 1.

I. Да се реализира клас **Device**, който съдържа информация за електронно устройство: марка, модел и цена. За този клас да се реализират:
   1) голяма четворка и конструктор с параметри;
   2) подходящи селектори и мутатори;
   3) функции за въвеждане и извеждане.

II. Производен клас **BrokenDevice**, която съдържа допълнителна информация за развалено устройство: симптоми и нужно време за поправка в брой дни. Да се допълнят функциите от а) - в) за производния клас.

III. Да се реализира клас **Service**, който съдържа два списъка: от здрави устройства и повредени устройства. За този клас да се реализират:
   1) голяма четворка;
   2) функции за въвеждане и извеждане;
   3) функция за добавяне на ново устройство;
   4) функция за приемане на заявка за повредено устройство;
   5) функция за прехвърляне на поправено устройство в списъка от здрави устройства;
   6) функция за отпечатване на всички устройства с даден симптом;
   7) функция за пресмятане на общата цена на всички наличните устройства;
   8) функция за пресмятане на евентуалните приходи за поправка на всички развалени устройства по дадена цена за работен ден за 1 устройство.

Задача 2.

I. Да се реализира абстрактен клас **Expression**, представящ израз, чиято стойност е число с плаваща запетая. Всеки израз да поддържа следните операции:
   1) double value () – стойност на израза
   2) void print () – печата информация за израза

II. Да се напишат три реализации на клас **Expression** – **Constant**, **Operation**, **Input**.
   1) Клас **Constant** да представя константен израз (число). Константата се задава при конструиране на обект. За този клас да се печата стойността на константата.
   2) Клас **Operation** да представя една от операциите +, -, *, / върху произволен брой изрази. Видът на операцията, масив от указатели към изрази и неговия размер се подават при конструиране на обекта. За този клас да се печата информация за всички изрази, участващи в операцията и самата операция.

Забележка: За операцията изваждане от първия израз се вадят всички останали, а за операцията деление – първият израз се разделя на произведението на останалите.

   3) Клас **Input** е израз, който зависи от вход от клавиатура. При изпълнение на value() се чете число от клавиатурата. За този клас да се печата само низът "Input".

III. Да се реализира функция double maxAll (Expression *arr[], int n), която намира максималната стойност измежду стойностите на изразите в масива arr.

Писмен изпит по обектно-ориентирано програмиране, 05.06.2011 г.

Вариант А

Задача 1. Да се реализира абстрактен базов клас **Set**, описващ множество от цели числа. Класът да има метод, който извършва проверка за принадлежност на дадено цяло число към множество.
Да се реализира производен клас **Multiple**, който представя множеството от тези цели числа, които се делят без остатък на някакво предварително зададено (в конструктора) цяло число.
Да се реализира и друг производен клас **Finite**, който представя крайно множество от до 100 числа чрез масив (който се задава в конструктора). Класът да поддържа добавяне и изтриване на елемент от множеството.
Да се реализира функция, която по зададен списък от множества (чрез масив от указатели) и някакво число проверява дали числото се съдържа в обединението на множествата от масива.

Задача 2. Да се дефинира шаблон на клас **Vector<T>** описващ колекция от последователни елементи от произволен тип **T**. Размерът на вектора се задава в конструктора му. За класа да се реализират:

1. оператори + и +=, събиращи покомпонентно елементите на вектора;
2. оператор за индексиране, даващ достъп за четене и писане до елементите на вектора;
3. необходимите методи от голямата четворка.

С помощта на шаблона **Vector<T>** да се реализира матрица, представена като вектор от вектори от числа с плаваща запетая.
Да се реализира функция, която по въведено от клавиатурата число N, създава две квадратни матрици с размерност N x N, въвежда елементите им от клавиатурата, събира ги и отпечатва резултата.

Задача 3. Да се реализира клас **SumAvgStat,** който представя сума на поредица от цели числа. При създаване на обект от класа, съответната му поредица съдържа само едно число, което се подава като аргумент на конструктора. За класа да се реализират следните методи:

- **sum,** който връща сумата на поредицата
- **add**, добавящ число към поредицата
- **del**, премахващ число от поредицата
- **num**, връщащ броя на числата в поредицата
- **average**, връщащ средното аритметично на числата в поредицата.

Функционалност извън тези 4 метода, като например съхраняване на отделните числа от поредицата, не е необходима.

Писмен изпит по обектно-ориентирано програмиране, 05.06.2011 г.

Вариант Б

Задача 1. Да се реализира абстрактен базов клас **Set**, описващ множество от цели числа. Класът трябва да има метод за проверка на дали цсичослен интервал с дадено начало и край е подмножество на множеството.
Да се реализира производен клас **Range**, който представя множеството от тези числа, които са извън даден целочислен интервал (началото и краят на интервала се задават в контруктора).
Да се реализира и друг производен клас **Finite**, който представя крайно множество от до 100 числа чрез масив (който се задава в конструктора). Класът да поддържа добавяне и изтриване на елемент от множеството.
Да се реализира функция, която по зададен списък от множества (чрез масив от указатели) и някакво число проверява дали числото се съдържа в сечението на множествата от масива.

Задача 2. Да се дефинира шаблон на клас **Vector<T>** описващ колекция от последователни елементи от даден тип **T**. Размерът на вектора се задава в конструктора му. За класа да се реализират:

1. оператор за индексиране, даващ достъп за четене и писане до елементите на вектора;
2. оператор < за сравнение на елементите на два вектора по големина. Приемаме, че A < B, ако всички елементи на A са по-малки от съответните елементи на B;
3. необходимите методи от голямата четворка.

С помощта на шаблона **Vector<T>** да се реализира матрица, представена като вектор от вектори от дробни числа.
Да се реализира функция, която по въведено от клавиатурата число N, създава две квадратни матрици с размерност N x N, въвежда елементите им от клавиатурата и проверява дали всички елементи на първата матрица са по-малки от съответните им елементи на втората матрица.

Задача 3. Да се реализира клас **PGMStat,** който представя произведение на поредица от строго положителни цели числа. При създаване на обект от класа, съответната му поредица съдържа само едно число, което се подава като аргумент на конструктора. Да се реализират следните методи на класа:

1. **prod,** който връща акумулираното до момента произведение
2. **add**, добавящ число към поредицата
3. **del**, премахващ число от поредицата
4. **num**, връщащ броя на числата в поредицата
5. **gMean**, връщащ средното геометрично на числата в поредицата.

Функционалност извън тези 4 метода, като например съхраняване на числата от поредицата, не е необходима.
Забележка: Приемете, че е разполагате с функция **root(*n*, *x*)**, пресмятаща корен ***n***-и от ***x***.

***Писмен* изпит по обектно-ориентирано програмиране,** 0*4.09*.2011 г. *Вариант 1*

Задача 1. А) Реализирайте абстрактен клас Question, който представя въпрос от тест (до 100 знака) и поддържа методите void askQuestion(), който извежда въпроса на екрана, въвежда отговор от потребителя и го запазва и bool checkQuestion(), който връща дали отговора на въпроса е въведен и верен. Реализирайте и конкретните наследници IntegerQuestion, който очаква като отговор цяло число, като верният отговор се задава в конструктора и OpenQuestion, който очаква като отговор произволен текст (до 255 знака), като верността на отговора се въвежда от екзаминатор, преглеждащ въпроса и дадения отговор.

Б) Да се реализира клас Test, представляващ списък от въпроси с произволна дължина. За класа да се реализират подходящ конструктор, метод void performTest(), задаващ последователно въпросите на потребителя и int numCorrectAnswers(), връщащ броя на верните отговори.

Бонус: реализирайте метод void shuffle(), който разбърква въпросите в даден тест на произволен принцип. Използвайте библиотечната функция int rand(), която връща случайно цяло число.

Задача 2. Да се дефинира шаблон на клас Relation<T>, който съдържа два обекта от тип T, наречени subject и object, и низ с произволна дължина relation, описващ връзката между тези обекти.

Пример: Relation<int> r1(2,6,“is smaller than”),r2(6,3,“is divisible by”);

За шаблона да се реализират голямата четворка и операция за отпечатване void print()

Пример: r1.print(): *2 is smaller than 6.*

За инстанцията на шаблона Relation<int> реализирайте и оператор за композиция * по следния начин.

Ако r = r1 * r2, то r.subject = r1.subject, r.object = r2.object,

r.action = r1.action r1.object “, which is“ r2.action

Пример: (r1*r2).print():*2 is smaller than 6, which is divisible by 3*

Композицията се допуска само ако r1.object == r2.subject, в противен случай резултатът е r1.

Упътване: Обръщението към ф-ята itoa(val,str,10) записва числото val в низа сочен от str.

Писмен изпит по Обектно-ориентирано програмиране, 07.09.2010 г.

Вариант А

Задача 1. Да се реализира клас EventLog, който съдържа списък от събития (описани от низове). Класът да предоставя възможност за регистриране на ново събитие и да има възможност да регистрира произволен брой събития. Да се реализират подходящи конструктори, деструктори и оператори за този клас, както и метод AddEvent, който регистрира събитие. Обърнете внимание, че е добре да пазите копие на текста на събитието вместо указателя, подаден от потребителя. Да се реализира и метод Print, който отпечатва на екрана всички събития.
Забележка. Не е нужно събитията да могат да бъдат редактирани и премахвани!

Задача 2. Да се реализира клас Building, който описва дадена сграда с височина (цяло число метри), площ (число с плаваща запетая в кв.м) и адрес (низ с произволна дължина). Да се реализира производен клас House, който допълнително задава брой етажи (цяло число) и име на собственик (низ с произволна дължина). За класовете да се реализират:

а) подходящи селектори и мутатори;

б) функции за въвеждане от клавиатура и извеждане на екрана;

в) функция, която по даден масив от къщи намира най-просторната къща, т.е. тази с най-голяма средна височина на етаж.

Задача 3. Да се реализира абстрактен базов клас IntSet, който описва следните операции върху изброимо крайно множество от цели числа:

- bool member(int x): проверява дали цялото число x е елемент на множество.
- int get(int i): връща i-тия елемент на множество. Индексацията на елементите е без значение.
- bool operator < ([попълнете правилния тип] s): проверява дали дадено множество е същинско подмножество на s.
- bool operator * ([попълнете правилния тип] s): проверява дали две множества имат непразно сечение.

Да се реализират наследници IntRange и ArraySet. Клас IntRange представя затворен интервал от цели числа. Краищата на интервала да се задават при конструиране на обекта. ArraySet е клас, поддържащ множество от максимум n цели числа, където n се задава по време на конструиране на обекта. Класът да поддържа следните операции:

- bool insert(int x): добавя числото x към множеството. Ако капацитетът е изчерпан, връща лъжа. Връща истина в противен слуай.
- bool remove(int x): премахва числото x от множеството. Ако елементът не съществува, резултатът е лъжа. Резултатът е истина в противен случай.

Да се реализира функция bool mon([попълнете правилния тип] sets[], int n), която получава масив от обекти (или указатели към обекти) множества. Масивът е с големина n. Функцията да проверява дали елементите на масива образуват строго монотонно растяща редица.

Септ. Сесия/2007

Задача 1.

а) Дефинирайте клас, представящ линейна функция от вида $y=ax+b$. Дефинирайте метод за изчисляване на стойността на функцията за конкретна стойност на x. Предефинирайте операторите събиране, изваждане и умножение така, че да намират сума, разлика и суперпозиция на линейни функции.

- Сумата на две фунцкции f и g дефинираме като такава функция h, за която $h(x)=f(x)+g(x)$ за всяко x.
- Разликата на две фунцкции f и g дефинираме като такава функция h, за която $h(x)=f(x)-g(x)$ за всяко x.
- Суперпозицията на две фунцкции f и g дефинираме като такава функция h, за която $h(x)=f(g(x))$ за всяко x.

Дефинирайте и метод, който по дадено число a връща нова линейна функция, която представлява «изместване» на функцията на разстояние a. Изместването на функцията f на разстояние a дефинираме като такава фунцкция h, за която $h(x)=f(x+a)$ за всички стойности на x.

b) Използвайки дефинирания в точка а) клас, дефинирайте клас дробно-линейна функция от вида $y=\frac{ax+b}{cx+d}$.

Задача 2.

Да се реализират следните класове:
Клас *връх*, който се описва с височина и име на съдържащата го планина.
Клас *туристически_връх*, който наследява *връх* и добавя името на върха.
Клас *военен_връх*, който наследява *връх* и добавя номер на котата на върха.
За класовете реализирайте подходящи конструктори, селектори и метод за извеждане на името на върха (за класа *военен_връх*, името е във формат “кота 54291”).

Задача 3. Да се създаде хетерогенен списък, елементите на който са символи, стекове и списъци от символи. Да се напише функция, която изтрива контейнерите, които съдържат само цифри ('0'...'9').