|  |
| --- |
| ЗАТВЕРДЖЕНО Наказ Міністерства освіти і науки, молоді та спорту України 29.03.2012 N 384 |

|  |
| --- |
| **Форма N Н-6.01**  Міністерство освіти і науки України  Вінницький національний технічний університет  Кафедра програмного забезпечення |
| КУРСОВА РОБОТА з дисципліни ” **Об’єктно-орієнтоване програмування** ” на тему: РОЗРОБКА ПРОГРАМНОЇ СИСТЕМИ ДЛЯ МОДЕЛЮВАННЯ  ОБ’ЄКТІВ ” ВУЛИЦІ, МІСЦЯ ДОЗВІЛЛЯ, ЛЮДИ”  З ВИКОРИСТАННЯМ UML ТА МОВИ ПРОГРАМУВАННЯ С# |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | |  | Студента (ки) \_\_1\_\_\_ курсу \_ групи \_ПІ-16  спеціальності 121 Інженерія програмного  забезпечення” Василевського В.  Керівник: доцент кафедри ПЗ, к.т.н., Д.І. Катєльніков Національна шкала \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Кількість балів: \_\_\_\_\_\_\_\_\_\_ Оцінка: ECTS \_\_\_\_\_\_\_ | | | Члени комісії | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (підпис) | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (прізвище та ініціали) | | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (підпис) | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (прізвище та ініціали) | |  |  |     м. Вінниця - 2017 рік  Міністерство освіти і науки України  Вінницький національний технічний університет  Факультет інформаційних технологій та комп’ютерної інженерії  ЗАТВЕРДЖУЮ  Зав. кафедри ПЗ, проф., д.т.н.  am2  \_\_\_\_\_\_\_\_\_\_\_\_\_\_ А.М. Пєтух  *(підпис)*  ”7” лютого   2017 р.    ІНДИВІДУАЛЬНЕ ЗАВДАННЯ  на курсову роботу з дисципліни ”Об'єктно-орієнтоване програмування”  студенту групи 1ПІ-16 Василевському В.О.  РОЗРОБКА ПРОГРАМНОЇ СИСТЕМИ ДЛЯ МОДЕЛЮВАННЯ ОБ’ЄКТІВ ” ВУЛИЦІ, МІСЦЯ ДОЗВІЛЛЯ, ЛЮДИ ” З ВИКОРИСТАННЯМ UML ТА МОВИ ПРОГРАМУВАННЯ С#  Вихідні дані:  - MS Visual Studio 2013, довідники з мови С# та Windows Forms.  Вимоги до курсової роботи:  Потрібно створити додаток Windows з графічним інтерфейсом. Клас мікрооб'єкта повинен містити не менше 4 елементів змінних і не менше 4 методів . Як мінімум одна змінна повинна бути типу int, одна – типу double і як мінімум одна – рядок (будь якої реалізації). В класі мікрооб’єкта повинен бути присутній конструктором по замовчуванню. Додати до класу мікрооб’єкта в якості елемента масив або посилання на невеличкий службовий об’єкт, щоб зробити необхідним глибинне копіювання. Реалізовано глибинне копіювання шляхом реалізації інтерфейсу ICloneable. Повинне бути продемонстроване використання глибинного копіювання. Перевантаження та використання унарного та бінарного операторів. Мікрооб'єкт: три графічних примітива. Більший макрооб'єкт: три графічних примітива. Менший макрооб'єкт: три графічних примітива. При натискуванні лівої кнопки миші на мікрооб’єкті він повинен ставати активним/неактивним. При натискувань клавіш-стрілок активні об’єкти/об’єкт повинні рухатись у вікні програми. При натискуванні клавіши Insert повинно з’являтись діалогове вікно, яке повинно визначати параметри створюваного мікрооб’єкта. Крім керуючого елемента Button у діалоговому вікні також повинні бути використані як мінімум три з наступного списку: TextBox, CheckBox, ListBox, RadioButton. В ньому повинна бути можливість обирати до якого з класів нащадків у ієрархії наслідування він належить. При натискувань клавіши Delete активні об’єкти повинні знищуватись. Якщо активного об’єкта нема – клавіша Delete ігнорується. Клавіша Esc повинна відміняти активацію обєкта. Одним з засобів повинна бути реалізована подвійна буферизація, яка повинна усунути миготіння при зміні зображення у вікні програми. Більший макрооб’єкт призначений містити всі мікрооб’єкти та всі менші макрооб’єкти. Графічне зображення більшого макрооб’єкта відображає лише певну його частину, яка не повинна перевищувати 25% його загального розміру. Менші макрооб’єкти повинні мати певні позиції в більшому макрооб’єкті і повинні відображатись на екраині. В більшому макрооб’єкті повинно міститись більше одного екземпляра меншого макрооб’єкта. Мікрооб’єкти можуть або належати одному (або більшій кількості) меншому макрооб’єкту, або не належати жодному. Повинна обов'язково бути релізовано можливість вилучити мікрооб'єкт з всіх менших макрооб'єктів, в результаті чого він не буде належати жодному меншому макрооб'єкту. Всі мікрооб'єкти та менші макроб'єкти повинні належати більшому макрооб'єкту і не повинні мати змоги вийти за його межі. Повина бути побудована ієрархія класів мікрооб'єктів, яка містить як мінімум три рівня наслідування, які можуть інстанціонуватись і зображатись на екрані. Зображення кожного рівня має відрізнятись від всіх інших хоча б одним графічним примітивом. Повинен бути реалізований автоматичний рух деяких мікрооб’єктів у програмі, при русі деякі мікрооб'єкти взаємодіють між собою (це має бути помітно візуально в тому сенсі, що або щось змінюється у їх зображенні або у характері їх руху). Повинен бути реалізований автоматичний рух деяких мікрооб’єктів у програмі, при русі деякі мікрооб'єкти заходять в менші макрооб'єкти та деякі виходять. Повинен бути реалізований автоматичний рух деяких мікрооб’єктів у програмі, при русі характер руху мікрооб'єктів повинен змінюватись при натискуванні певної клавіші клавіатури або при виборі команди меню. Повинні бути продемонстровано використання упаковка/розпаковка (boxing/unboxing). Повинні бути продемонстровано икористання multicast-делегат. Повинні бути продемонстровано використання асинхронний виклик делегата. Повинні бути продемонстровано використання тип доступу protected. Повинна бути запрограмована серіалізація/де-серіалізація всіх об’єктів у файл. Серіалізація/де-серіалізація обов’язково повинна зберігати не тільки власне інформацію про стан макро- та мікро-об’єктів, але й про їх позицію на екрані. При серіалізації/де-серіалізації обов’язково повинні використовуватись діалогові вікна, щоб запитати у користувача ім’я файлу та його розташування на диску (наприклад можна використовувати системні діалогові вікна OpenFileDialog та SaveFileDialog). Запрограмовані такі відносини між класами та об'єктами, які можуть бути наглядно проілюстровані на діаграмах агрегації. Запрограмовані такі відносини між класами та об'єктами, які можуть бути наглядно проілюстровані на діаграмах композиції. Запрограмовані такі відносини між класами та об'єктами, які можуть бути наглядно проілюстровані на діаграмах асоціації. Запрограмовані такі відносини між класами та об'єктами, які можуть бути наглядно проілюстровані на діаграмах кооперації. Запрограмовані такі відносини між класами та об'єктами, які можуть бути наглядно проілюстровані на діаграмах послідовності. Запрограмовані такі відносини між класами та об'єктами, які можуть бути наглядно проілюстровані на діаграмах стану. В програмі повинно бути продемонстровано використання поліморфізму. В програмі повинно бути продемонстровано використання інтерфейсу, створеного користувачем. В програмі повинно бути продемонстровано використання механізми динамічного приведення типу: ключові слова as та is.  Зміст ПЗ до курсової роботи  Індивідуальне завдання  АНОТАЦІЯ  ВСТУП  1 АНАЛІЗ СУЧАСНОГО СТАНУ ПИТАННЯ ТА ОБҐРУНТУВАННЯ ЗАВДАННЯ НА РОБОТУ  2 РОЗРОБКА ІНТЕРФЕЙСУ ПРОГРАМИ ТА ПРИКЛАДІВ ВИКОРИСТАННЯ  3 РОЗРОБКА ДІАГРАМ КЛАСІВ, ОБ’ЄКТІВ ТА СТАНУ  4 РОЗРОБКА ПІДСИСТЕМИ ГРАФІЧНОГО ВІДОБРАЖЕННЯ  5 ВИКОРИСТАННЯ ЗАСОБІВ ПРОГРАМУВАННЯ WINDOWS FORMS  6 РОЗРОБКА ПІДСИСТЕМИ СЕРІАЛІЗАЦІЇ/ДЕСЕРІАЛІЗАЦІЇ ДАНИХ  7 КЕРІВНИЦТВО КОРИСТУВАЧА  ВИСНОВКИ  ПЕРЕЛІК ПОСИЛАНЬ  Додатки (за необхідністю)  Дата видачі ”7” лютого 2017 р. Керівник \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  *(підпис)*  Завдання отримав \_\_\_\_\_\_\_\_\_  *(підпис)* |

ВСТУП

Почуття – невід’ємна частина людського життя. Разом з емоціями та рисами характеру вони створюють те, що ми називаємо внутрішнім світом людини. Інтернет трактує почуття як специфічні людські, узагальнені переживання, ставлення до людських потреб, задоволення або незадоволення яких викликає позитивні або негативні емоції - радість, любов, гордість або сум, гнів, сором тощо. Стає зрозуміло що не усі почуття, як і емоції є не лише позитивними, а й негативними, тобто такими, які спричинюють людині внутрішні тривоги, переживання, а інколи й страждання, проте це є частиною людського буття.

За однією з концепцій давньокитайської натурфілософії -- “Інь-янь” – не може бути нічого абсолютного поганого, або доброго, тому в кожному негативному є позитивне, й навпаки. Транслюючи цю думку на життєвий шлях, можна з впевненістю сказати, що на кінці свого шляху, людина пройде скрізь моменти горя і щястя.

Ніхто не буде сперечатись, що як казав Вергілій “Кохання переважує все”, адже усі знають що кохання це виключно світле почуття, здатне зробити людину кращою. Тому сьогодні переважна більшість самотніх молодих людей прагнуть знайти своє кохання, створити пару й прожити щасливе життя зі своєю половиною.

Мало хто замислюється про негативні аспекти настільки прекрасних почуттів. Насамперед неможливо бути впевненим у взаємності, адже інша людина має власні відчуття й емоції, закохавшись не завжди можна з абсолютною впевненістю сказати що ти зможеш побудувати міцні відносини зі своєю парою. У сьогоденні ми маємо багато прикладів, коли невзаємні почуття руйнують людину й її внутрішній світ, що є зовсім негативним проявом, здавалось б виключно світлих намірів. Нікому не подобається розуміти невзаємність, але наводить все більше й більше таких прикладів.

Закоханість, відчуття кохання – усе це, проходили велика кількість людей. Мабуть усі скажуть ще це й справді щось неймовірне й прекрасне. Досі навіть науковці, філософи, мислителі не можуть зійтись у думці що ж таке кохання, як воно виникає і який вплив воно має на життя того хто любить і життя того кого люблять, але заперечувати факт того що взаємне кохання є одним з найсвітліших почуттів в душі людини не може ніхто.

Саме тому в якості теми курсової роботи було обрано розробку програмної системи моделювання діяльності людей під час їх дозвілля, зустрічей з іншими людьми й закоханості.

# АНАЛІЗ СУЧАСНОГО СТАНУ ПИТАННЯ ТА ОБҐРУНТУВАННЯ ЗАВДАННЯ НА РОБОТУ

* 1. **Предметна область**

**Людина** (англ. *human*, нім. *mensch*) — жива, наділена інтелектом істота, суб'єкт суспільно-історичної діяльності і культури. Відмінними рисами людини, які визначають її унікальність у Всесвіті, є її свідомість, здатність мислити і здатність до здійснення вільного вибору, приймати відповідальність за власні дії; наявність моральних суджень. Описуючи людину, відмічають її біологічну непристосованість, відсутність спеціалізації його органів для якого-небудь конкретного простого тваринного існування, здатність створювати знаряддя праці і користуватися ними та вогнем, дар мови, пластичність поведінки. Невідомо жодної іншої істоти, яка має вищі емоції, традиції, здатність мислити, стверджувати, заперечувати, рахувати, планувати, фантазувати, знає про свою смертність, кохає в справжньому сенсі цього слова, має почуття гумору, здійснює свої задуми, відтворює наявне і створює щось нове. Людина вивчає і змінює як навколишній світ, так і самого себе, розбудовує культуру і створює власну історію. Сутність людини, її походження і призначення, місце людини у природі є основними питаннями філософії, релігії, науки і мистецтва. Виходить що люди бувають зовсім різні, хоча науковці й намагаються відзнайти ідеальну класифікацію людських характерів, але все це не більше ніж приблизне виокремлення рис, які разом частіше за інші є в людини.

Розрізняють не менш чотирьох підходів до розгляду терміну людина:

* людина розумна як біологічний вид;
* людина як щось, що виходить за рамки живого світу і в значній мірі протистоїть йому;
* людина як людство в цілому;
* людина як індивід, особистість.

**Відпочи́нок**, **дозвілля** — проведення часу, метою якого є відновлення нормального стану організму. Слід виділити загальноприйняті місця, такі як кафе, ресторани, ігрові майданчики, бари, розважальні центри, аквапарки, міські парки, тощо.

Люди повсюди комунікують і взаємодіють між собою, для зустрічей, окрім бажання зустрітись, необхідне заздалегідь зазначене, або ні, місце. Такі локації можуть стати як місцем зустрічі незнайомих людей, друзів, закоханих, родини, так і просто місцями для релаксу, відпочинку, використання локації визначає сама людина. Зустрічі можуть бути очікувані й навпаки, з новими людьми, або вже зі знайомими, бажана, або ні. Кожна з них та їх комбінації, разом з психо-емоційним станом людини, створюють очікувану від людини поведінку під час зустрічі, що зумовлюю хід, розвиток й завершення комунікації

* 1. **Існуючі реалізації**

Програми, які розкривають сутність людського спілкування, зустрічей та дозволяють промоделювати поводження людини залежно від їх стану під час комунікацій достатньо. Існують як звичайні ігри для браузерів(flash-ігри), так і повноцінні проекти, а також чати, форуми, які зазвичай задовільняють потребу в спілкуванні.

Перш за все існують чати та форуми які дозволяють людині поділитись враженнями, запитати поради, допомогти, або висловити свою думку с приводу певного питання. Вони стали невід’ємною частиною спілкування в мережі. Головним недоліками є подекуди відсутність модерації, а на популярних форумах, нездатність швидко організувати модераторську роботу, тому ви можете довго чекати на відправку свого повідомлення. А також низький рівень користувацького інтерфейсу, що не полегшує адаптацію нових користувачів.



Рисунок 1.1 - Приклад інтерфейсу загальнодоступного Вінницького форуму

Іншим шляхом комунікування, проте з набагато кращим розважальним елементом є ігри, наприклад гра що поширюється просторами інтернету й дозволяє користувачам взаємодіяти між собою з елементами інтертейменту – “Аватарія”



Рисунок 1.2 - Початковий інтерфейс гри “Аватарія”

Ця гра дозволяє людині знаходити друзів в ігровому форматі, проте вимагає постійного підключення до інтернету й ви не можете бути впевнені що розмовляєте з тим, за кого людина по сторону екрана себе видає.

Також шляхом самопізнання, перевірки своїх комунікативних навичок, ідеями для покращення цих здібностей є програми-тестування.

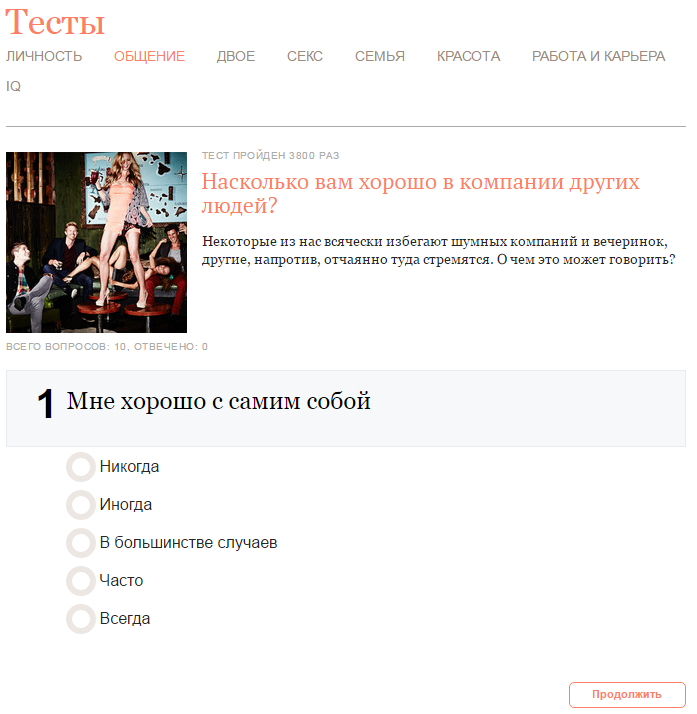


Рисунок 1.3 - Тест наскільки вам зручно в компанії

Вони поширюються через мережу Інтернет, є локальними програмами, або проектами, які вимагають підключення до мережі для відображення результату. Явним недоліком є невідповідність результатів до реального життя, адже тестування людських навичок неможливе шаблонно.

**1.3 Розробка технічного завдання на роботу**

Для нормального виконання програми необхідно наступне апаратне забезпечення:

1. Операційна система: Windows 7, 8, 10 , миша, клавіатура
2. Процессор: 2.0 GHz
3. Оперативна пам’ять: 4 GB ОЗУ
4. ВІдеокарта: 1 GB of Video RAM
5. Вільне місце в HDD: 400 Мб.

**1.4 Обґрунтування вибору мови програмування**

Мова С# має багато переваг у порівнянні з іншими популярними мовами програмування, які вказані у таблиці 1.1:

Таблиця 1.1 – Перваги мови програмування С#

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | Збирач сміття | Препроцесор | Підтримка виключень | Кроссплатформеність | Об'єктно-орієнтована мова | Ручне керування пам’ятю | Локальні функції |
| Java | + | - | - | + | **+** | - | **-** |
| C++ | **-** | **+** | **+** | **+/-** | **+** | **+** | **+** |
| C# | **+** | **+** | **-** | **+** | **+** | **+** | +/- |

Збирач сміття є важливою складовою мови C# оскільки дає змогу не замислюватись про звільнення виділеної попередньо пам’яті, а отже неможливо помилитись із цим.

Препроцесор дає можливість переглянути код до початку компіляції, що може дати попередження про фатальні помилки.

Відпрацьована підтримка виключень в мові C# дозволяє коректно відображати помилки під час виконання програми.

Мова С# є засобом об'єктного програмування, нової методики проектування та реалізації програм, яка в поточному десятилітті, найімовірніше, замінить звичайне процедурне програмування. Головною метою творця мови доктора Бьерна Страустрапа було оснащення мови С# конструкціями, що дозволяють збільшити продуктивність праці програмістів і полегшити процес оволодіння великими програмними продуктами.  
 Абстракція, реалізація, спадкування й поліморфізм є необхідними якостями якими володіє мова С#, тому за численні переваги також перед мовами Java та C++ було вибрано цю мову для створення програмного продукту.

**1.5 Висновки**

Дослідивши проблему фактичної відсутності програмних продуктів, які б демонстрували важливість комунікації та людських стосунків в житті кожної персоны, було вирішено створити програму, яка могла б в певній мірі відобразити складність та розповсюдженість взаємозв’язків між людьми в їх вільний час.

Також, упродовж виконання курсового проекту, було вивчено об'єктно-орієнтовану мову програмування C#, а також написання консольних проектів та проектів з використанням WinForms, було досліджено переваги та недоліки мови С#.

3 РОЗРОБКА ДІАГРАМ КЛАСІВ, ОБ’ЄКТІВ ТА СТАНУ

3.1 Діаграми класів

На діаграмах класів показано різноманітні класи, які утворюють систему і їх взаємозв’язки. Діаграми класів називають “статичними діаграмами”, оскільки на них показано класи разом з методами і атрибутами, а також статичний взаємозв’язок між ними: те, яким класам “відомо” про існування яких класів, і те, які класи “є частиною” інших класів, — але не показано методи, які при цьому викликаються.

3.1.1 Діаграми наслідування

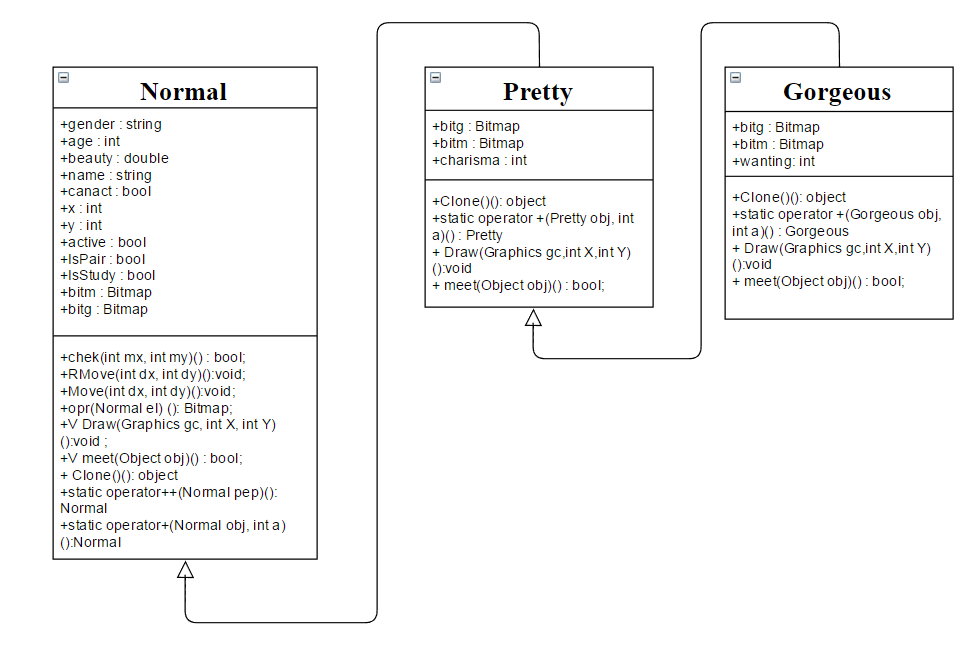
Наслідування є однією з фундаментальних основ об’єктно-орієнтованого програмування, у якому клас “отримує” всі атрибути і операції класу, нащадком якого він є, і може додавати власні атрибути і операції. 

Рисунок 3.1 – Діаграма наслідування

Значення змінних:

* gender – стать мікрооб'єкта;
* age – вік мікрооб'єкта;
* beauty – привабливість мікрооб'єкта;
* name – ім’я мікрооб'єкта;
* canact – можливість руху;
* x – координата розміщення мікрооб’єкта;
* y – координата розміщення мікрооб’єкта;
* active – активність мікрооб'єкта;
* IsPair –знаходження у парі;
* IsStudy – знаходження у режимі навчання;
* bitm – Bitmap мікрооб'єкта;
* bitg – Bitmap мікрооб'єкта;
* charisma – харизма мікрооб'єкта;
* wanting – бажання знаходитись в парі мікрооб'єкта;

“V” перед методами класу означає, що функція віртуальна.

3.1.2 Діаграми агрегації

Місця дозвілля можуть містити в собі певну кількість людей, проте при вилученні людей, це не нашкодить більшим мікрооб’єктам. Таке відношення є агрегацією.

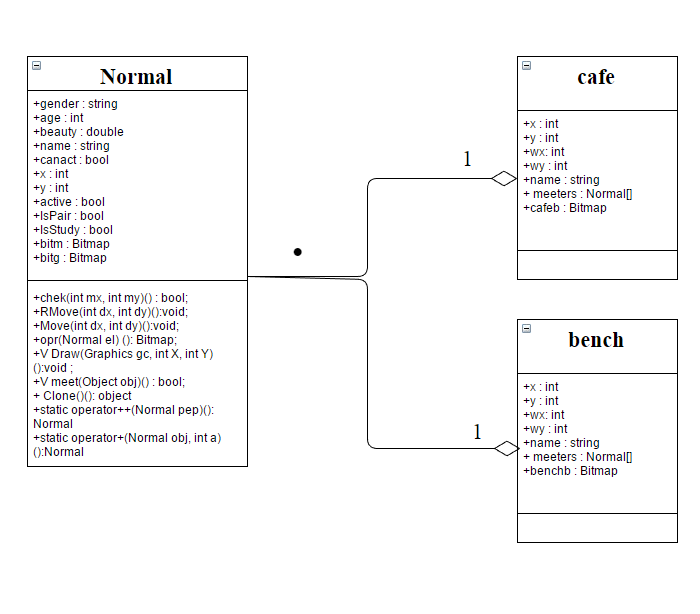


Рисунок 3.2 – Діаграма агрегації

3.1.3 Діаграми композиції

Координати x та y є невід’ємною частиною класу і він перестане існувати, якщо її вилучити. Таке відношення називається композицією.

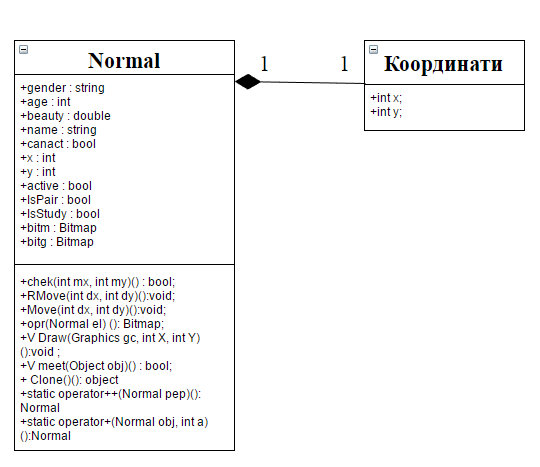


Рисунок 3.3 – Діаграма композиції

3.1.4 Діаграми асоціації

При асоціації встановлюється такий зв’язок, при якому елементи (класи) не залежать один від одного, тому можуть вільно існувати окремо. У нашому випадку прикладом може слугувати зв’язок двох мікрооб’ктів. Об’єкти за певних умов можуть взаємодіяти, або не взаємодіяти взагалі.

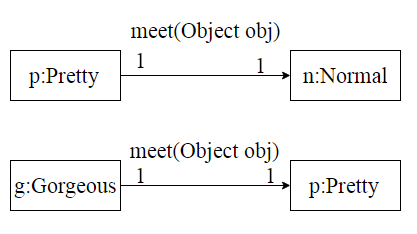


Рисунок 3.4 – Діаграма асоціації

3.2 Діаграми кооперації

На діаграмі показано шлях запису у файл за допомогою окремого класу saved.

Позначення, що використовуються в діаграмах кооперації:

о:C – об'єкт із власним іменем о, екземпляр класу С.

:C – анонімний об'єкт, екземпляр класу С.

о: (чи просто о) — об'єкт-сирота з власним іменем о.

о/R:C - об'єкт із власним іменем о, екземпляр класу С, що грає роль R.

/R:C - анонімний об'єкт, екземпляр класу С, що грає роль R.

о/R - об'єкт-сирота із власним іменем о, що грає роль R.

/R— анонімний об'єкт і одночасно об'єкт-сирота, що грає роль R.

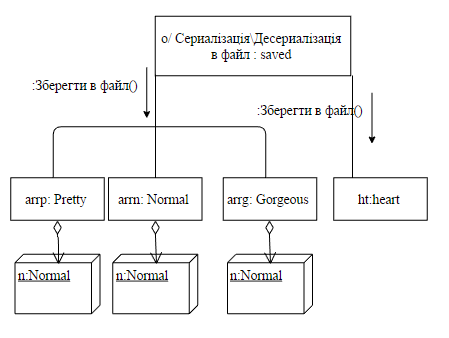


Рисунок 3.5 – Діаграма кооперації

3.3 Діаграми послідовності

Діаграма послідовності відображає взаємодії об'єктів впорядкованих за часом. Зокрема, такі діаграми відображають задіяні [об'єкти](http://uk.wikipedia.org/wiki/%D0%9E%D0%B1%27%D1%94%D0%BA%D1%82_%28%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F%29) та послідовність відправлених повідомлень.

В якості прикладу відобразимо асинхронний виклик multicast-делегата movement для об’єктів n та p та массиву arrn, а також синхронне додавання нового елементу в масив arrn.

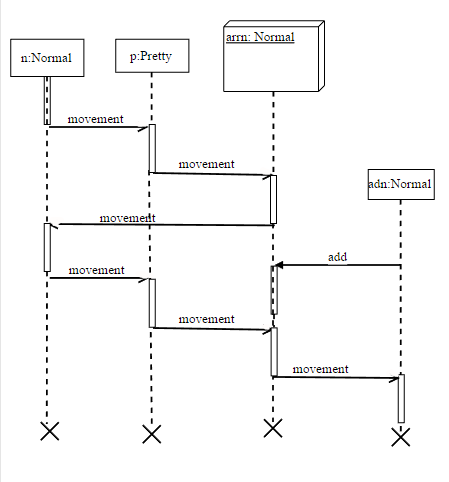


Рисунок 3.6 – Діаграма послідовності

3.4 Діаграми стану

На діаграмі стану зображено алгоритм прийняття рішення мікрооб’єктом чи вступати в стосунки.

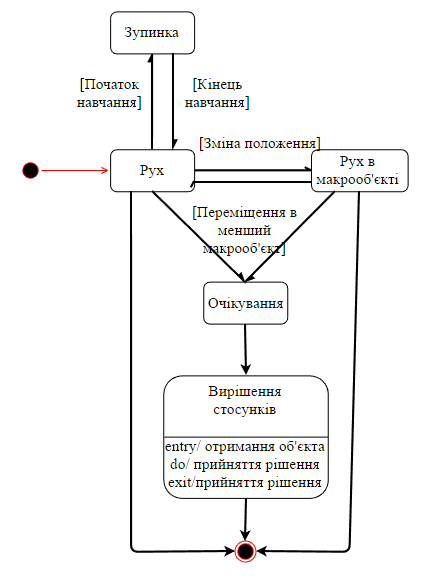


Рисунок 3.7 – Діаграма стану