Kursnamn: Objektorienterad Programmering Grund
Klass: Handelsakademin OOP med AI-Kompetens
Termin: HT-22

Produkt-Automat

INLEDNING

Bakgrundsbeskrivning	Uppgiften går ut på att skapa en C# applikation i konsol som uppfyller nedanstående krav. Uppgiften inkluderar också design i form av UML diagram som beskriver relationen mellan klasser i applikationen
Varför ska ni utföra detta arbete?	Syftet är att öva i att bygga upp en applikation efter en beskrivning samt också modellera ett UML klassdiagram efter samma beskrivning. Ger också övning i samspel mellan olika klasser samt koncept som Arv & Interface etc.
Vad ska ni leverera?	Ni skall leverera en färdig applikation i C# som kan exekveras i VS. Ett UML Klassdiagram, inkluderat attribut och metoder för varje klass, sparat i lämpligt bild format (png eller jpeg).

ER PROJEKTUPPGIFT

Vad ska ni göra?	Läs igenom beskrivningen nedan som beskriver hur applikationen skall fungera vid avslutat arbete. Baserat på beskrivningen skall ni rita ett UML Klassdiagram som beskriver relationen mellan de klasser som behövs. När Klassdiagrammet är färdigt, utgå ifrån det för att programmera applikationen. Fokusera på att skapa dina klasser först och sedan bygg upp ett program-struktur runt klasserna. Använd lämpligt verktyg för att rita UML klassdiagramet (draw.io eller LucidChart t,ex) och VS för att programmera applikationen
Hur löser ni uppgiften?	Förstå och tolka beskrivningen. Markera olika key-words i texten för att underlätta. Substantiv med en färg, adjektiv med en annan färg och verb med en tredje färg. Detta hjälper er att kunna separera på vad som blir klasser, attribut och metoder vardera.
Struktur för arbetet Vid grupparbete: arbetsfördelning och tidsplan	Uppgiften är en individuell uppgift.

INLÄMNING OCH REDOVISNING

Inlämning	Projektet skall vara klart och färdigställt för bedömning senast Fredagen den 18 November
Redovisning	Denna uppgift skall inte Redovisas Denna uppgift behöver inte en skriftlig rapport.

BEDÖMNING OCH ÅTERKOPPLING

Bedömning sker med följande betygskriterier	**För godkänt (G) på projektarbetet skall följande krav uppfyllas:** Ett UML Klassdiagram som visar relationen mellan klasserna i applikationen. En applikation baserat på nedanstående beskrivning som uppfyller följande krav: ● Användaren skall kunna se produkter i en meny och skall kunna välja en produkt via konsolen.

- Vid val av produkt skall användaren kunna se produktens beskrivning innan användaren väljer att köpa varan. Användaren skall kunna acceptera köpet eller välja att gå tillbaka till menyn.
- Vid köp av produkt skall produkten köpas och användas. Användaren tas tillbaka till menyn vid avslutat köp.
- Ett sista menyval som avslutar programmet.

För väl godkänt (VG) på projektarbetet skall dessutom följande krav uppfyllas:
Allting under Godkänt skall uppfyllas.

Applikationen skall även uppfylla följande krav:

- Ett nytt menyval; att kunna mata in pengar till automaten. Användaren skall kunna se hur mycket pengar som finns i automaten.
- Vid köp av produkt så skall en kontroll ske; att användaren har matat in tillräckligt med pengar till automaten. Om inte så stoppas köpet.
- När användaren väljer att avsluta programmet skall kvarstående pengar i automaten returneras. Skriv ut hur mycket pengar som returneras samt i vilken valör.

Återkoppling

2 December

BESKRIVNING

Beskrivningstext:

I konsol, skriv ett program som presenterar en automat för användaren. Denna automat skall erbjuda flera olika produkter. Användaren skall kunna välja att se information om en enskild produkt samt välja vilken produkt som skall köpas. Tänk på att varje produkt används olika, tex en mat-produkt skall ätas, en kläder-produkt skall bäras på kroppen, en dricka-produkt skall drickas etc. Varje produkt skall ha sin egen Klass. Alla dessa produkt-klasser skall ha samma Interface samt ärva från samma abstrakta klass. Detta Interface skall innehålla Description(), Buy() och Use(). Den abstrakta klassen skall innehålla attribut för namn, kostnad och beskrivning. Självklart kan fler metoder och attribut läggas till om du känner de behövs.

Varje produkt skall ha en kostnad. Användaren skall ha en plånbok med pengar. När programmet startar har användaren 10 st av varje valör (dvs 10st 1-kronor, 10st 5-kronor, 10st 10-kronor, etc). Användaren måste mata in pengar till automaten innan användaren kan köpa en produkt. Naturligtvis skall användaren själv kunna bestämma hur mycket som skall matas in samt vilka valörer. Vid avslut skall användaren få tillbaka eventuell växel som finns kvar i automaten som inte har blivit spenderad, naturligtvis avrundad uppåt. (Om användaren har matat in 2st 10-kronor så får användaren tillbaka en 20kr sedel.) Plånboken skall ha en egen klass som uppdaterar användarens pengar.