



# HÖGSKOLAN I BORÅS

## **P0 - Projekt Kick-off**

*Instruktioner för genomförandet*

2.5hp, grupparbete

**Grundläggande Programmering i C# NGC011 VT23**

# Handledare i projektet

Antonios Tsertsidis (ANTS), grupp 1-20

[antonios.tsertsidis@hb.se](mailto:antonios.tsertsidis@hb.se)

Karl-Axel Zander Persson (KAZA), grupp 21-40

[karl-axel.zander\\_persson@hb.se](mailto:karl-axel.zander_persson@hb.se)

Fereshta Westin (FEWE), grupp 41-60

[fereshta.westin@hb.se](mailto:fereshta.westin@hb.se)

Rikta frågor rörande *arbetsprocessen med er projekt* till er tilldelade handledare, och i första hand på de veckovisa handledningstillfällena. Generella frågor rörande uppgiften, examination, handledningstillfällen osv kan skickas till mig (Karl-Axel)

Administrativa frågor rörande gruppindelningarna tas med kursansvarig

Anna Palmquist (ANP)

[anna.palmquist@hb.se](mailto:anna.palmquist@hb.se)

# Vad projektet innebär

- Framtagning av en (.NET Core) konsol-applikation i C#
  - Designa systemet (*lagom* omfattande, se exempelprojekt senare)
  - Implementera (kod) – Här är fokus i denna kurs!  
Applikationen blir troligen någon form av register, där man (åtminstone minst) kan lägga till, ta bort, sortera och söka efter saker (data).
- Ni väljer själva vad applikationen skall handla om och till viss del hur den skall fungera
- Görs som grupparbete: ni flesta bör redan vara färdigindelade i grupper om 3 sedan kursintroduktionen. *Om inte så lös det snarast (eget ansvar!), kontakta Anna om du önskar hjälp med att hitta en grupp att ansluta till*
- Ska vara ett **unik**t projekt, dvs er grupp skall inte avsiktligt göra samma sak som andra grupper projektidéer

# Exempel på lämpliga projekt *välj något nedan, eller hitta på något eget!*

- Företagskundregister
- Medlemsregister
- Startlistor för travtävlingar
- Register över (fotbolls-)spelare
- Kompiskatalog
- Fastighetsförsäljning
- Campinggästregister
- Lagersystem för artiklar
- Bokhandelssystem
- Program för val av viner till mat
- Enkelt spel med lagring av resultat
- Filmregister  
(titel, längd, genre, mm)
- Utlåning/uthyrning av något  
(vilket, vem, när)
- Register över låtlistor/låtar i Spotify
- Receptregister
- Släktforskningsregister
- Kennel
- Kostnadsövervakning för hobby  
(ex folkrace)
- Butiksdatasystem
- Matchmaking (kärlek, jobb)

# Syftet med projektet

Är inte att ni *nödvändigtvis* måste skapa något jätteinnovativt- och användbart...

Men tanken är oavsett att ni genom att genomföra projektet på ett bra sätt ska:

- Få förståelse för hur kodning fungerar generellt och bli bekväma med att hantera utvecklingsmiljön (Visual Studio)
- Få förståelse för allmänna kodkoncept som loopar (iteration), villkor (selektion) och metoder
- Lära er att förstå och kunna granska kod som andra har skrivit
- Bli bra på “kodformalia”, t.ex användande av lämpliga (beskrivande) variabelnamn och att kunna kommentera koden på ett lagom sätt
- Träna på att utveckla programvara i grupp (*ofta inte helt lätt!*)

...Och därmed

- Bli väl förberedda på kommande programmeringskurser samt
- Klara tentamen

# Minimikrav för projektet

*checklista att stämma av med!*

Detta **måste** ingå i projektet och förhållas till


- Iterationer/loopar (while, for)
- Lämpligt valda datatyper för variabler
  - Exempelvis skall numeriska värden ha numeriska datatyper (och inte string...)
- Metod för att visa er data (ex. "VisaLista()")
- Metoder för att lägga till och ta bort element i/ur en vektor/array. *Använd ej dynamiska listor, se nedan*
- Metoder för sortering och sökning
- Metoder för läsning och skrivning av/till (txt) fil
  - Förslagsvis användas till "persistent lagring" av data likt en databas mellan programkörningar
- Felkontroller på inmatning (även kallat validering)
  - Så att användaren inte kan krascha programmet, vare sig avsiktligt eller oavsiktligt
- Koden skall genomgående vara uppdelad i lämpliga metoder (tumregel: en uppgift - en metod)
- Några specifika saker som gör er kod ej OK:
  - Användandet av "goto". Det är programlogiken som skall styra flödet
  - Anropandet av Main/programloopen inifrån egenskapande metoder
  - Användandet av dynamiska listor (t.ex "List<T>"), som har inbyggda funktioner för sortering etc. Ni ska istället bara använda vektorer/arrayer i pedagogisk syfte i denna projektuppgift
- Koden skall vara välkommenterad (se Canvas-sidan:
- En tillhörande (uppdaterad) projektbeskrivning



Kodkommentering - riktlinjer

# Projektbeskrivning

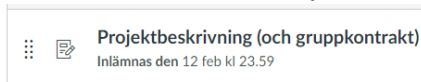
Utöver koden så ska tillhörande dokumentation skrivas i form av en projektbeskrivning, och ska innehålla:

- Bakgrundbeskrivning: *Vad är systemet till för? I vilket sammanhang skall det användas? Av vilka typer av användare?*
- Beskrivning i textform av programmets funktion (som en "teknisk manual", vilka menyer finns, hur funkar dom, osv)
- Schematiskt beskrivning över programmets funktion (=ett flödesschema, se Canvas-sidan:  [Ritverktyg till flödesscheman](#))
- En lista med de lagrade data i programmet (t.ex. *filmtitel*, som skall vara en "string". *produktionsår*, som ska vara en "int". Osv)

Dokumentationen behöver inte vara jätteformell och nödvändigtvis inte längre än 1x A4 (utöver era flödesscheman)

Projektbeskrivningen lämnas in två gånger:

- Först en "tidig" version innan ni börjar koda, så som ni (efter bästa förmåga) tror att systemet kommer se ut och fungera. Detta är **valfritt** och är ett extra tillfälle ni kan få feedback på att ni är "på rätt spår". Sista datum är **12/2** att lämna in ifall ni vill nyttja detta, se särskild Canvas-Uppgiftssida:



I samband med denna inlämning så **rekommenderar** vi att ni även läser igenom och lämnar in ett påskrivet "spelregler-gruppkontrakt" (se mall i Canvas). För er egen skull att bättre kunna hantera diverse sociala situationer om något sådant motförmodan uppstår i gruppen under arbetsperioden

- Sedan en uppdaterad andra version med "hur det faktiskt blev" när ni är klara med kodandet. Lämnas in tillsammans med koden inför aktuellt examinationstillfälle (**obligatoriskt**)

# Examination

OBS nytt upplägg från VT23-

Projektet examineras muntligt (redovisning och "utfrågning"), med individuell bedömning för varje enskild student. Redovisningen utförs dock gruppvis. Betygskala U/G. Ert projektmaterial ska lämnas in i förhand i samband med anmälan till aktuellt examinationstillfälle (3st extillfällen erbjuds, se schemat för datum)

Anmälan görs till examinationstillfället genom att lämna in ert projekt (kod + projektbeskrivning) under Uppgifter i Canvas, deadline är ett par dagar innan redovisningsdagen (se schema)



Projekt - anmälan och inlämning av projektmaterial till (muntlig) examination 1 [2.5hp, betygskala U/G]

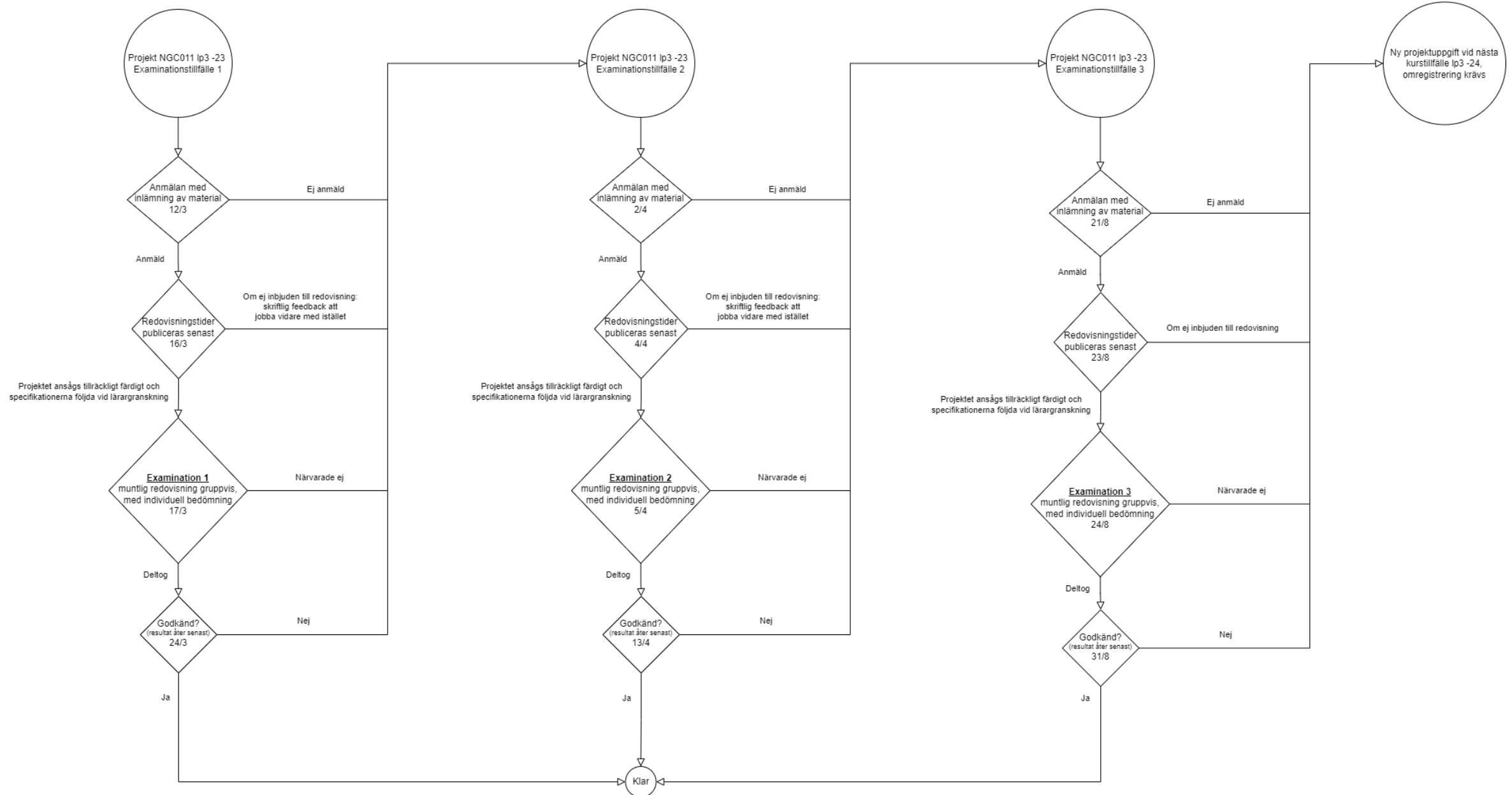
Tillgänglig till mar 12 at 23:59 | Inlämnas den 12 mar kl 23:59 | 2,5 poäng

Vi granskar sedan raskt efter deadline ert inlämnade material och verifierar att det är *tillräckligt färdigt & välgjort, och har följt specifikationen*. OBS att detta är i sig dock ej någon delbedömning som sedan direkt påverkar sitt omdöme på den kommande muntliga examinationen.

Är fallet så, så blir man inbjuden till examinationen (=man blir aviserad med en redovisningstid). Om inte, så får man istället skriftlig handledningsfeedback om vad gruppen behöver jobba vidare med, och hänvisas till ny anmälan vid nästa examinationstillfälle



# Överblicksbild för projektexaminationen för kursomgången i ett flödesschema



# Anmälan med inlämning av projektmaterial

Detta skall skickas in i samband med i "anmälan"



Projekt - anmälan och inlämning av projektmaterial till (muntlig) examination 1 [2.5hp, betygskala U/G]

Tillgänglig till mar 12 at 23:59 | Inlämnas den 12 mar kl 23:59 | 2,5 poäng

- Hela Visual Studio-projektmappen till er applikation (dvs inkl. sln-fil, textdatafil och exe-filen), packat i ett zip-arkiv
- Projektbeskrivning (inklusive flödesschema), lämna in som pdf

*Tips (och viktigt om ni är ovana att använda zip-arkiv!): test-skicka zip-filen innan inlämning till en annan dator med Visual Studio och test-öppna+kör därifrån! Kontrollera att det fortfarande fungerar som det ska.*

# Examination – muntliga redovisningen

20 minuter redovisning/"utfrågning" per grupp effektiv tid, och görs inför er handledare. Görs i grupprummen D308, D310 och D312 (*kan vid extraordinära omständigheter flyttas till Zoom som backup*). Redovisningstider skickas ut senast dagen innan via Anslag i Canvas (ni bokar & väljer inte tid själva för detta). *Preliminära* redovisningstider (fm/em) efter gruppnummer finns dock redan nu upplagt för er framförhållnings skull, se denna Canvas-sida:

Grupparbete och gruppindelning

Genomförandet blir efter dessa punkter:

- Legitimering (glöm inte legitimation!)
- Ni börjar *kort* med (max 5min) att köra & visa programmet, samt projektbeskrivningen. Ni kan t.ex berätta lite hur ni tänkte under arbetsprocessen, vad som var lätt/svårt, visa något intressant avsnitt i koden, osv. *Ni kan förutsätta att handledaren är lite bekant med ert projekt sedan innan och gå ganska "rakt på sak"*
- Ni får sen först någon enstaka fråga som ni får diskutera och svara på tillsammans i gruppen
- Varje gruppmedlem får avslutningsvis sedan ett antal frågor den får individuellt svara på/redogöra över

Ni behöver inte ta med någon laptop att koppla in i projektorn, utan ert iförväg inlämnade material kommer tas fram på grupprums-datorn (som har Windows som OS) av handledaren via Canvas, och ni får styra denna dator under redovisningen. *Observera alltså att ni INTE kan finputs ert projekt efter anmälan och komma med en uppdaterad version att använda under redovisningen. Inga undantag på detta, lika för alla! Stödanteckningar på papper eller via laptop fås tas med om man vill.*

Redovisningen spelas in via video i grupprummet, och blir underlaget för bedömning. *Inspelningar sparas inte längre än nödvändigt efter kursen i enlighet med högskolans riktlinjer för detta*

# Examination – muntliga redovisningen forts.

Omdömen/betyg sätts ej direkt, utan presenteras inom en vecka efteråt (via Canvas)



Projekt - anmälan och inlämning av projektmateriäl till (muntlig) examination 1 [2.5hp, betygskala U/G]

Tillgänglig till mar 12 at 23:59 | Inlämnas den 12 mar kl 23:59 | 2,5 poäng

Får man underkänt så får man en skriftlig bedömningskommentar och vad man behöver träna mer på förståelsemässigt om innehållet i sitt projekt. *Vid godkänt så får man oftast ingen bedömningskommentar alls utan bara betyget, förkännedom*

Efter ett examinationstillfälle kan det bli så att 1-2 studenter i en grupp av 3 blir underkända (eller inte kunde delta, av godtycklig anledning). Isåfall får den/de underkända anmäla sig till nästa examinationstillfälle och komma på ny muntlig redovisning/examination, som genomförs på samma sätt. Inlämning av projektmaterialet ska göras även då i samband med "anmälan", och man får till dess om man vill jobba vidare med projektet till en uppdaterad version, för sitt lärande skull. Men inte nödvändigtvis, var det "OK" förra gången för att bli inbjuden till redovisningen, så blir det "OK" igen också)

# Examination – muntliga redovisningen, förberedelser

Aktuella grupprum som kommer används till redovisningarna kan bokas av studenter "till vardags" (i KronoX). Vill man bekanta sig med rummet och den stationära datorn där så kan man göra det. Är man Mac-användare så är det även viktigt att göra sig bekant med Windows-versionen av Visual Studio så redovisningen flyter på bra för er

Ni kan med fördel specifikt förbereda den inledande delen där ni "fritt uppvisar" ert projekt, så att denna görs på effektivt/intressant sätt och att ni samtidigt försöker uppvisa kunskaper/"att ni har lärt er". *OBS igen att detta dock är ett mycket kort inslag, 5min. Lägg inte all tid på bara exempelköra programmet*

Men i övrigt så förbereder ni er bäst att lyckas bara med att ni alla i gruppen gör er väl insatta i alla delar av koden och projektbeskrivningen. Samt även att träna på att använda rätt programmatiska begrepp när man pratar om koden, så att ni lättare kan uppvisa er förståelse. Att säga t.ex "här tilldelas variabeln x till värdet y" istället för "här blir x lika med y", osv.

Det kan vara så, och är helt naturligt, att en i gruppen har arbetat *mer* med någon del i projektet/koden än de andra. Men det blir inte en ursäkt under den muntliga examinationen att man inte individuellt kan besvara "vilken specifikt ställd fråga om vad som helst" rörande projektet/koden. Så tips desto mindre en gruppmedlem har jobbat med att ex. tillverka en viss del av koden under arbetet, desto viktigare att den ordentligt pluggar på i efterhand inför redovisningen hur den koden faktiskt funkar i detalj!

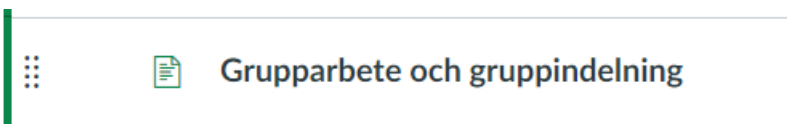
Exempel på *karaktär av frågor* som ni kan få vid "utfrågnings"-delen (att ibland begäras att svaras på tillsammans med stöd från varandra, eller helt individuellt)

- "Vad gör raden kod X för något?"
- "Förklara hur denna metod X fungerar, gå igenom rad för rad vad som händer"
- "Visa exakt vart den motsvarande saken X i flödesschemat görs i koden?"
- "Hur tänkte ni när ni valde att designa systemet/programmet på detta sätt X?"
- *Om något påvisas inte fungerar helt korrekt/optimalt* - "hur skulle man kunna lösa eller förbättra sak X?"

# Arbetsordning och handledning

- Handledning i “storgrupp” i datasal. Här brukar det bli generella frågor om hur man angriper uppgiften och kommer igång, som andra grupper också har nytta av att höra. Vi inleder även med en liten genomgång där vi levererar lite sammanställda tips vi tror många har nytta av. 90-min pass
  - P1 fre 3/2 – Databehov och strukturer
  - P2 fre 10/2 – Modularisering (procedurer och funktioner)
  - P3 fre 17/2 – Strukturerad programmering
- Per-grupp-handledning i Zoom enligt bokning i Canvas-kalendern (15min). Ha era frågor & funderingar redo. *Se till att ni provat att dela skärm innan så att ni vet att det funkar som det ska!*
  - P4 fre 24/2 – Statiska och dynamiska data
  - P5 fre 3/2 – Sökningar och sorteringar
  - P6 fre 10/3 – “Sista handledning inför examination 1”
  - P7 fre 30/3 – “Kompletterande handledning inför examination 2”

Canvas-sida för salar och Zoom-länkar:



*Tillfällena P1-P5 har ett informellt “tema-område”, vilket fungerar i praktiken som rekommenderad arbetstakt (“detta bör ni ha hunnit funderat på/kommit till för att ligga i fas”).*

# Handledning, förväntningar

- Vi handledare finns här för att förtydliga projektinstruktioner, diskutera val av angreppssätt, och i viss mån hjälpa er praktiskt när ni kör fast! Vi erbjuder relativt mycket handledningstid i denna kurs, dock i praktiken förutsatt att man drar igång med sitt arbete redan "vecka 1" och försöker ligga i fas! Vi kan inte kompensera missade eller dåligt utnyttjade handledningstillfällen
- Vi försöker undvika att "skriva koden åt er" - det blir en ineffektiv björntjänst i både tidsutnyttjande och i ert lärande
- Mot slutet av projektet, så undanbes frågor av typen *"om vi lämnar in projektet precis som det är nu, kommer det garanterat bli 'godkänt' då?"* Sådana saker måste ni alltid försöka avgöra själva i slutändan!

# Hjälpmedel

- Direkt samarbete mellan grupper (eller utomstående personer) är ej tillåtet. Dock tillåtet att diskutera projektanvisningarna och angreppssätt på ett generellt vis
- Annars inga begränsningar
- **FAQ:** *Får man använda AI (t.ex ChatGPT) för projektet?* Ja, och det kan vara ett bra verktyg i programmering för generera kodsnuttar ("kickstarta" sin programidé), rätta kod som inte fungerar, få teoretiska saker förklarade ("personalized tutor"), m.m. Det är bra att *"work smarter not harder"*!  
Observera dock att examinationen är muntlig, så undvik den "lockande genvägen" att lämna in kod som ni inte "förstår" själva! Det visar sig sen ändå hur man egentligen ligger till kunskapsmässigt gentemot lärandemålen i kursplanen. Man tar sig inte igenom kursen (och definitivt inte kommande påbyggande kurser) om man faktiskt inte ser till att lära sig programmering *från grunden*



# Detaljer och lite tips!

# Lite kort om namngivning

- Vi kommer vara noga med att ni har bra namn på era variabler och metoder!
- Tänk på stora och små bokstäver – den vanligaste namnkonventionen i C# är:
  - Variabler skrivs med "camel notation", dvs "minVariabel"
  - Metoder och klasser skrivs med "Pascal notation", dvs "MinMetod()"
- Detta gör att det blir mycket enklare för andra att följa er kod
- Se även till att strukturen på koden är läsbar och utan onödigt klutter

# Det är så lätt att "fixa det senare"...

- Men, ni gör det mycket enklare för er själva att koda " snyggt" från början!

```
public Class Class1
{
    int a;

    public void    b(int c)

    a = c;
    {
        Console.WriteLine(a);
    }
}
```

Eller...

```
public Class MinKlass
{
    public string minVariabel;

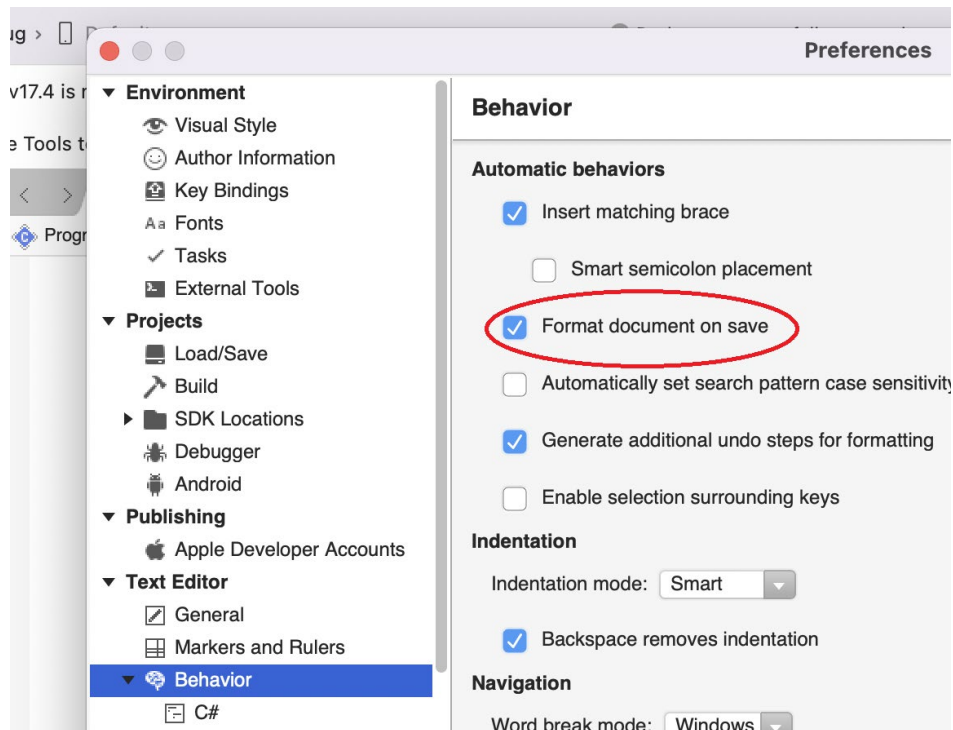
    public void skrivUt(int antalGångar)
    {
        for (int i = 0; i < antalGångar; i++)
        {
            Console.WriteLine(minVariabel);
        }
    }
}
```

# Tips – autoformatering av koden i Visual Studio (2019)

*Nyttja dessa så håller ni er kod alltid i “perfekt stil”!*

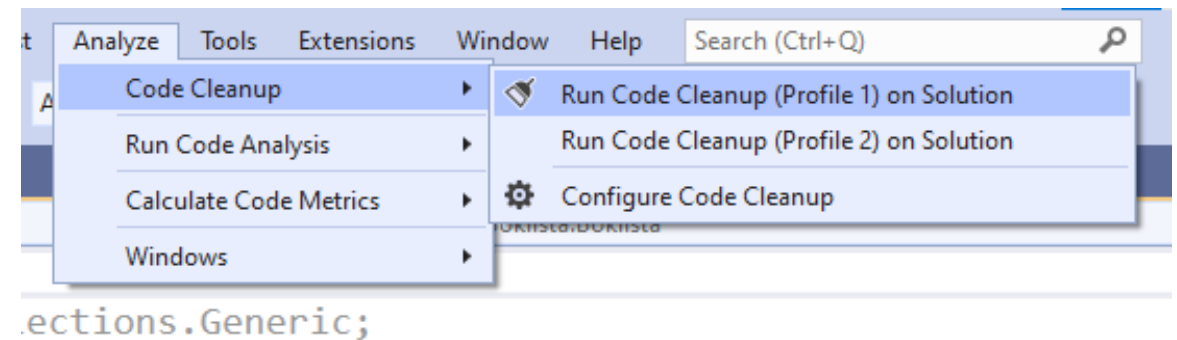
## Mac

*Kodformateringen körs varje gång man sparar*



## Windows

*Kodformateringen behöver initieras manuellt*

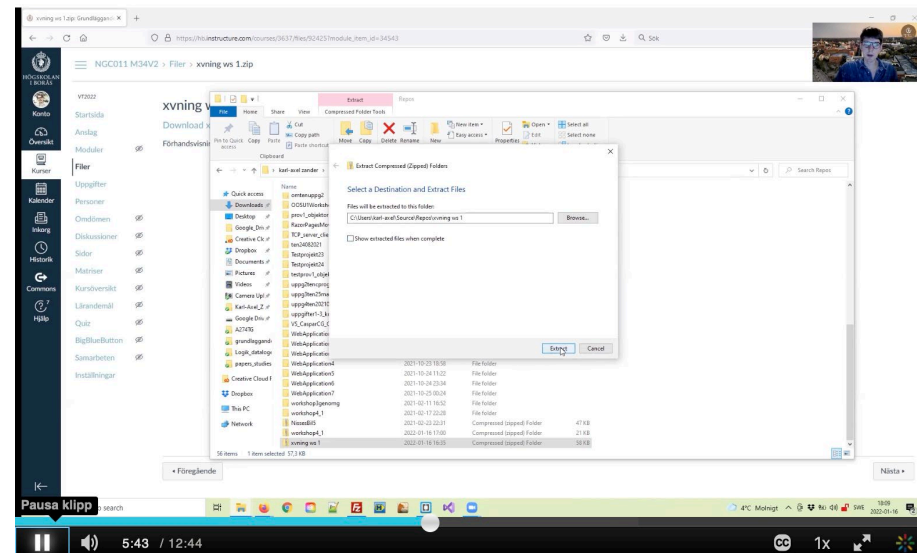
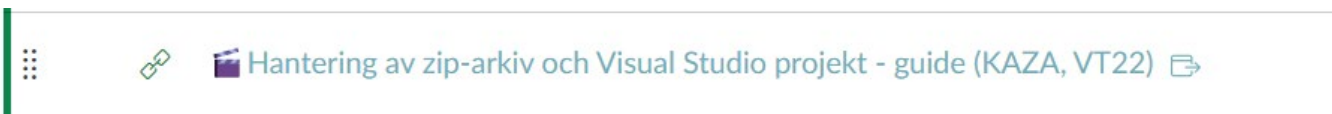


# Att spara och lämna in sitt Visual Studio-projekt

Det händer tyvärr för ofta att någon grupp lämnar in fel version av koden, eller gör något tokigt vid packandet till en ziparkiv-fil (det sättet/format ni är ombudade att lämna in i Canvas genom) så att innehållet sen saknas/blir korrupt vid öppnande. Dubbelkolla innan inlämning!

*Se videon som går igenom detta:*

*Visar både för Windows/Mac*



- Namnge era versioner smart när/om ni gör större ändringar vid något tillfälle under arbetet! Istället för Kompisregister\_1, Kompisregister\_2 etc eller Kompisregister\_ny, Kompisregister\_nyare ... så skriv Kompisregister200214, Kompisregister200216, Kompisregister200216b etc. Då ser ni ju också direkt vilket datum respektive version är ifrån

# Koddelning och samarbetsverktyg

Vid arbete med projektet i grupperna så är det bra att från början etablera en rutin för att dela koden med varandra. Annars får ni lätt problem om någon gruppmedlem ex. blir oväntat frånvarande.

Realtidssamarbetsverktyg (flera personer kan arbeta parallellt med koden) kan även vara värt att titta på och komma igång med, så alla i gruppen kan vara med och koda "hands-on" hela tiden

Tips på tjänster/verktyg för detta finns på Canvas-sidan:



Koddelning och samarbetsverktyg

*OBS dock att vi i denna kursen inte har någon möjlighet att erbjuda någon teknisk support på någon av alla dessa nämnda*

*Vi hjälper till med Visual Studio 2019, andra program/tjänster/verktyg får ni felsöka på egen hand ifall ni stöter på problem*

# Se till att koda mycket!

- Ju mer tid ni lägger på arbete med koden (både kvalité och volym), ju lättare kommer redovisningen gå, samt även tentan! Se till att alla i gruppen aktivt är med och skriver kod! Denna kurs är en viktig grund, och byggs vidare på senare i utbildningen (*Objektorienterad systemutveckling 1* m.fl för SV & DE)
- Blir ni färdiga ”för tidigt”, utöka gärna funktionaliteten lite i programmet! Se det som kombinerat redovisning- och tentaplugg för att fästa kunskaperna i långtidsminnet

# Kennelregistret

Pedagogisk genomgångsminiserie om utvecklingsstegen mot ett enkelt konsolregisterprogram

*Konkreta tips för hur man kan komma igång, visandes ett exempel som sannolikt kommer likna er projektidé*

📎	KennelRegister_P1_Databehov_och_strukturer.png
📎	<a href="#">KennelRegister P1 - Databehov och strukturer (KAZA, VT22)</a>
📎	KennelRegister_P2_modularisering.png
📎	<a href="#">KennelRegister P2 - modularisering (KAZA, VT22)</a>
📎	KennelRegister_P3_strukturerad_programmering.png
📎	<a href="#">KennelRegister P3 - strukturerad programmering (KAZA, VT22)</a>

Del 2 & 3 publiceras längre fram i kursen

