Vejledning / Råd og vink

Valgfags-bekendtgørelsen

Programmering C

Undervisningsministeriet
Afdelingen for gymnasiale uddannelser
2010

Programmering C – Valg
Vejledning / Råd og vink

Afdelingen for gymnasiale uddannelser 2010

Alle bestemmelser, der er bindende for undervisningen og prøverne i de gymnasiale uddannelser, findes i uddannelseslovene og de tilhørende bekendtgørelser, herunder læreplanerne. Denne Vejledning/Råd og vink indeholder forklarende kommentarer til nogle af disse bestemmelser, men indfører ikke nye bindende krav. Desuden gives eksempler på god praksis samt anbefalinger og inspiration, og den udgør dermed et af ministeriets bidrag til faglig og pædagogisk fornyelse. Citater fra læreplanen er anført i kursiv.

Indhold side

1. Identitet og formål

(der henvises til læreplanen)

2. Faglige mål og fagligt indhold

2.1 Faglige mål

Eleverne skal kunne:
– *redegøre for programmering som planlægning af en computers aktiviteter, herunder interaktion med omgivelserne*
– *læse enkle programmer og redegøre for deres funktionsmåde*
– *rette og tilpasse enkle programmer*
– *anvende eksisterende programdele og biblioteksmoduler i arbejdet med at programmere et fungerende system*
– *demonstrere kreativitet og systematik i programmeringsprocessen*
– *løse en enkel problemstilling gennem udviklingen af et program.*

Hverdagen indeholder mange programmerbare komponenter, som interagerer med omgivelserne og de vil ofte være en egnet indfaldsvinkel i undervisningen. Udgangspunktet er, at alle programmerbare komponenter, herunder computere, kun gør hvad de får besked på. Nogle af disse komponenter kan anvendes til at bibringe eleven en erkendelse af, hvad brugerflader er og hvordan de er tænkt, ligesom selve programmernes virkemåde kan undersøges og diskuteres. Et simpelt eksempel kan være en cykelcomputer, et vækkeur eller en mobiltelefon. Mulighederne er mangfoldige.

Som udgangspunkt bør eleverne forstå, at programmering indeholder nogle almene begreber, som alle er indeholdt i de fleste programmeringssprog. De almene begreber kan overføres til andre sprog, som eleverne senere kan prøve kræfter med.
Det afgørende er, at eleverne bliver fortrolige med et sprog. Det er ikke et mål at eleverne skal kunne redegøre for- eller skelne mellem flere programmeringssprog og programmeringsmiløer. Afhængig af valgte temaer og problemstillinger, eller i forbindelse med supplerende stof, kan det være hensigtsmæssigt at inddrage andre programmeringssprog og programmeringsmiljøer og deres anvendelsesmuligheder.
Ved valg af programmeringsmiljø bør man gøre sig overvejelser angående tekstbaserede og grafisk programmerede miljøer, da der er forskelle i omfanget af integrerede værktøjer. Arbejdes evt. med flere sprog og/eller miljøer skal eleven begrunde dette ud fra disses egnethed til løsning af konkrete opgaver.

De fleste elever har kun lidt eller ingen erfaring med at programmere. Udgangspunktet er derfor enkle programmer, hvor eleven kan læse og redegøre for funktionsmåden. Når eleven løser opgaver eller arbejder med et projekt, kan det ligeledes være nyttigt at anvende eksisterende programdele og rette disse til. (se også 3.1 Didaktiske overvejelser)

I de fleste programmeringssammenhænge anvendes biblioteksmoduler. Det er en nødvendig del af programmeringsfaget at brugen heraf indarbejdes. Eleverne kan via internettet og andre steder selv finde biblioteksmoduler, der kan bidrage til at løse det problem, de arbejder med. De mere erfarne elever kan selv strukturere deres kode gennem funktionel opdeling i metoder, og gøre den mere overskuelig ved at lave biblioteksmoduler og/eller klasser.

Gode arbejdsvaner er også i programmering en afgørende hjælp til at holde styr på og bevare

overblikket. Undervisningen skal derfor give eleverne redskaber, der kan støtte dem i en målrettet og systematisk arbejdsproces hen imod løsningen af en given problemstilling. Her vil pseudokode, flowcharts og lignende redskaber kunne støtte eleverne i udviklingen af dele af koden.

I slutningen af forløbet skal eleven selvstændigt kunne bruge disse redskaber på simple problemstillinger. Hvis man ikke gennemtænker programmet, vil man let støde i problemer med at løse problemstillingen, og vil i bedste fald få en løsning, der måske nok fungerer, men som tit vil være svære at vedligeholde og rette i.
En vigtig del af elevernes udvikling er at forstå, at programmeringen først starter, når man har planlagt, hvad programmet skal kunne, hvordan det skal se ud, og hvordan tingene skal implementeres. Og det foregår ikke ved at programmere uden plan.

Kreativitet forbindes oftest med kunstneriske produkter. Men arbejdet med programmering og programmerbare løsninger rummer også mange kreative udfordringer så som ideer til løsningsmuligheder, måder at implementere et problem i koden og arbejde sig frem mod mere raffinerede løsninger. Her anvendes teknikker, som kendes fra projektorienteret undervisning i andre fag. Grafisk design og efterfølgende programmering af brugerflader er dels en kreativ proces, hvor eleverne kan udfolde egne ideer, dels en systematisk proces, hvor der arbejdes med generelle principper for dialogdesign og brug af gængse principper for brugervenlighed og grafisk fremstilling.

I slutningen af forløbet skal eleven selv kunne løse en enkel problemstilling gennem udviklingen af et program. Her skal indgå hensigtsmæssige arbejdsgange og abstrakte beskrivelser af programmeringen. Som underviser skal man hjælpe eleverne med at finde det niveau, den enkelte har, så de er i stand til selv at at løse en given problemstilling. Denne problemstilling vil oftest danne udgangspunkt for et eksamensprojekt eller et forudgående projekt.

2.2 Kernestof

Kernestoffet er:
– programmeringssprog
– elementer i programmeringssprogets opbygning såsom data- og kontrolstrukturer
– programmers regelbundne opførsel ud fra programmets enkelte elementer
– programmers interaktion med omgivelserne
– programdele og biblioteksmoduler
– arbejdsgange i programmeringsprocessen
– abstrakte programmeringsbeskrivelser og dokumentation.

Det er tilrådeligt, at der kun vælges ét primært programmeringssprog af hensyn til den pædagogiske tilrettelæggelse og overskueligheden for eleverne. I en differntieret undervisnings situation kan der vælges et sekundært sprog til nogle elever. I et tværfagligt samarbejde med eksempelvis matematik, kan Mathcad eller andre matematikprogrammer godt fungere sideløbende med det valgte sprog.

Det valgte sprog bør give mulighed for at arbejde med grundlæggende data- og kontrolstrukturer, dvs. datatyper, variabelbegreb, sekvenser, løkker og forgreninger, samt aritmetiske beregninger, så eleven har en klar forståelse af disse strukturer. Begreber som metoder bør også indgå, og klasser kan også indgå på et elementært niveau, hvis det valgte sprog indbyder til det.

For den enkelte elev er det centralt at erkende, at programmerbare enheder kun gør det, de er programmeret til. Gode eksempler fra hverdagen kan være med til at anskueliggøre dette. Et vigtigt

element er at kunne læse og forklare enkle programmers virkemåde, så de kan analysere egne programmer og undgå fejl. Eleven skal altså kunne skelne mellem et programs statiske opbygning og dynamiske opførsel, som f.eks. giver sig udtryk i den korrekte implementering af en løkke, og dens virkemåde under programafvikling. Dialogen mellem lærer og elev og eleverne imellem er her et vigtigt element, der kan støtte eleverne i senere abstrakte beskrivelser af programmer, ligesom bl.a. pseudokode og flowchart støtter udviklingen af programstrukturen.

Arbejdsgangene er en vigtig del af programmeringsprocessen. Nogle gode vaner vil også støtte eleven i at beskrive programudviklingen i journaler, så de bliver i stand til at reflektere over- samt dokumentere deres arbejde på en præcis og forståelig måde. Journaler er dermed også med til at tydeliggøre for eleven, at det er vigtigt under programmeringsfasen, at føre notater/logbog over de processer, de gennemarbejder. Det er en hjælp at vænne eleverne til at skrive velordnede programmer med fornuftige sigende variabelnavne, anvende indrykninger der angiver strukturen i programmet, samt at kommentere programmet, så andre dels kan læse det, men også vil være i stand til at arbejde videre på koden. Her er eksemplets magt en god inspiration.
Undervisningseksemplerne skal altså være formet over den samme læst.

2.3 Supplerende stof
Det supplerende stof skal opfattes som fordybelsesområder der ligger i forlængelse af den øvrige undervisning. Eksempelvis kan nævnes programmering af en mobiltelefon, hvis eleverne har arbejdet i et Java baseret miljø; design af databaser, hvis eleverne har arbejdet med internetbaseret programmering i eksempelvis PHP eller ASP.

Programmering har mange muligheder for at indgå i tværfaglige sammenhænge med fag som matematik, fysik, elteknik, kommunikation/it, hvor der naturligt kan indgå programmeringsprocesser.
Det faglige samspil i studieretningen kan tilgodeses også gennem valg af supplerende stof i programmeringsfaget.
Udviklingen inden for it går hurtigt og mange nye områder ser dagens lys hvert år. Mobiltelefoner, netværksteknologier, 3D mv. er nogle af disse. I den udstrækning der er praktisk mulighed for det, kan det supplerende stof også kombineres med studiebesøg hos it-virksomheder, laboratorier og videregående uddannelsesinstitutioner.

(paradigmatiske eksempler kan ses på fagets side på EMU:
http://www.emu.dk/gym/hhxhtx/pm/index.html)

3. Tilrettelæggelse

Som studieretningsfag indgår faget på linje med de øvrige fag i planlægningen af den samlede studieretning, og, som nærmere uddybet i afsnit 3.4, er der adskillige muligheder for at faget kan virke katalyserende for undervisningen i flere fag.
Som valgfag er de tværfaglige samarbejdsmuligheder af naturlige grunde færre, men en differentieret undervisningsplanlægning kan gøre det muligt at finde tværfaglige samarbejdspartnere. Endelig er formaliseret tværfagligt samarbejde ikke altid en nødvendig forudsætning for, at den enkelte elev kan erkende et samspil mellem fagene. Gennem undervisningsdifferentiering kan eleven, eller en gruppe af elever, arbejde med tværfaglige opgaver og projekter.

3.1 Didaktiske overvejelser
Elevforudsætninger

Erfaringsmæssigt har eleverne vidt forskellige programmeringsmæssige forudsætninger ved starten af forløbet, og undervisningsdifferentiering er et vigtigt redskab til at fastholde en tilstrækkelig individuel progression.
Differentieringen kan eksempelvis ske gennem udstrakt inddragelse af eleverne i undervisningen gennem valg af emner, opgaver, eksempler, elevoplæg mv. Elever med erfaring kan udnyttes som en vigtig ressource for undervisningen i programmering. Der bør tilsvarende arbejdes indgående med individuel evaluering.
Kun undtagelsesvis har elever programmeringsmæssige færdigheder med sig fra folkeskolen. Derimod kan eleverne forventes at have rimelige generelle it-bruger kompetencer, både fra folkeskolen og fra de første års undervisning på gymnasialt niveau.
De mere erfarne elever har altså hovedsageligt selvlærte kvalifikationer med relation til programmering, ofte uden en systematisk tilgangsvinkel. Det medfører, at også de mere erfarne elever, kan have brug for hjælp til strukturering af deres viden og færdigheder. De er derfor ingenlunde selvhjulpne, men har et anderledes behov med hensyn til vejledning og progression i forløbet.

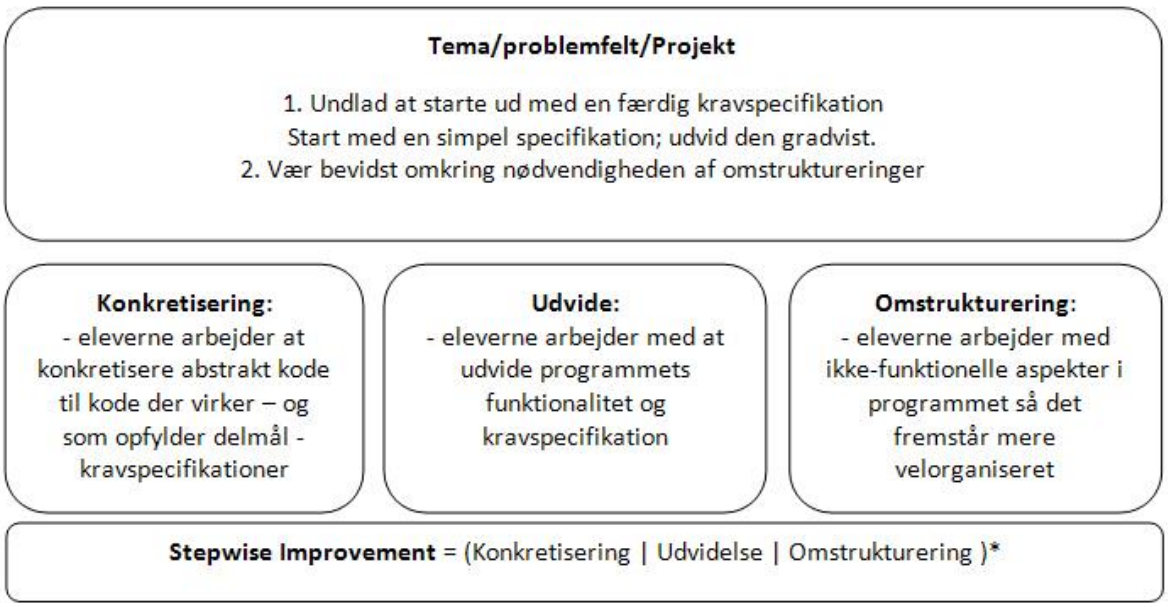
Synliggør de faglige mål
Med udgangspunkt i de meget forskellige elevforudsætninger er det formålstjenligt at indlede undervisningen med at skabe klarhed og overblik hos eleverne om fagets mål og indhold. Såvel øvede som mindre øvede elever kan have udbytte af dette forløb, fx med henblik på faglig strukturering, bevidstgørelse af eleverne om deres medansvar for udbytte af forløbet og justering af den efterfølgende undervisning. Det er vigtigt at den allerførste præsentation af faget lægger vægt på succesoplevelser, eksempelvis at afslutte et lille men fungerende program.

Valg af undervisningsmaterialer
Programmering udgør for mange elever et helt nyt stofområde, og kernestoffet er derfor af betragteligt omfang set i relation til fagets tidsramme. Dertil kommer at en del af undervisningen ikke benyttes til eksplicit gennemgang af kernestoffet, men derimod til procesorienterede forløb. Det er afgørende at der eksisterer tilstrækkeligt med opslagsbøger, elektroniske opslagsværker og andre hjælpekilder, eleverne kan anvende under deres individuelle arbejde.
Det vil være en fordel at basere undervisningen på et enkelt programmeringssprog, specielt af hensyn til de elever der ikke har programmeret før. Det er en fordel for alle parter, hvis man tager afsæt i de samme problemstillinger, og alle arbejder i samme udviklingsmiljø. Eleverne får dermed fordelen af at kunne inspirere og hjælpe hinanden.
Det er altså bedst at tage diskussionen om valg af sprog med det samme, og erfaringsmæssigt vil modstanden mod et sprog forsvinde, når eleverne får en god struktureret indføring i sproget og dets metoder, og en fornuftig arbejdsplatform.
Afhængigt af det sprog og programmeringsmiljø undervisningen centreres om, kan det være sparsomt med velegnede lærebøger. Muligheden for velegnet undervisningsmateriale kan blive et rent praktisk argument for valg af sprog og programmeringsmiljø. Ofte vil læreren dog være henvist til selv at skrive noter til brug for bestemte lektioner og forløb, eller alternativt sortere og udvælge fra den store mængde af webbaseret materiale.
Især ved brug af webbaseret materiale skal man være opmærksom på, at der ofte vil være behov for individuel tilpasning. Selv om alle elever må forventes at være rimeligt sprogligt funderede i både dansk og engelsk, kan den store mængde af nye fagtermer være et argument for at lægge vægten på dansksprogede tekster. Især ved projektarbejde, og løsning af særlige problemer, vil det være relevant at inddrage engelsk materiale og fx hente kommenterede kodeeksempler fra engelsksprogede websteder.

Valg af programmeringsmiljø
Udvælgelsen af et primært programmeringsmiljø bestemmes i første omgang af lærerens egne kompetencer. Dertil kommer skolens økonomiske prioriteringer, idet licensbaseret software udgør en betragtelig omkostning. Der eksisterer også mulighed for at benytte sprog og programmeringsmiljøer, der er gratis at bruge på uddannelsesinstitutioner. Læreren må desuden foretage en vurdering af, hvilke sprog der lettest kan give begynderen en stringent indføring i generelle principper og metoder. Det er selvsagt en væsentlig faktor for mange elevers indfrielse af de faglige mål. En anden relevant faktor er hensynet til planlagt tværfagligt arbejde.

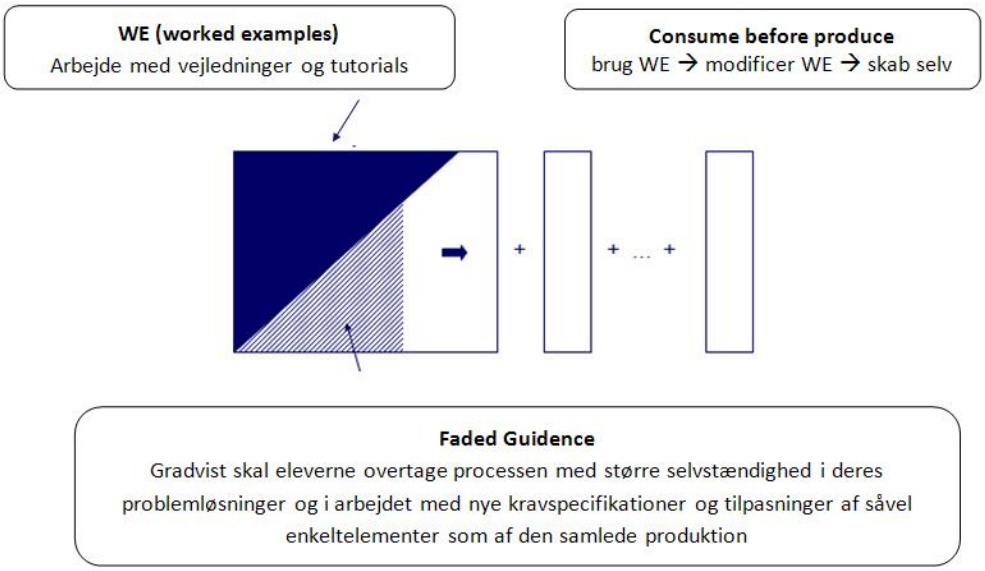
Udgangspunkt i hverdag og teknologi
I nogle tilfælde kan man ikke inden for fagets rammer nå at arbejde med konkrete programmerede genstande, men ofte kan man arbejde med forsimplede problemstillinger simuleret i det primære udviklingsmiljø, eller inddrage programmerede genstande i diskussioner uden dette skal udmøntes i programmer.
Det er en god idé på et tidspunkt at lade eleverne arbejde direkte med identifikation af programmerbare objekter og spekulere over disses regelbundne opførsel. Her kan både være tale om teoretiske øvelser og en undersøgelse af programmerbare objekter på egen skole. Mulighederne er mangfoldige.
Valg af problemstillinger med udgangspunkt i elevernes hverdag kan have stor betydning både for elevernes engagement og for den faglige overførselsværdi til andre områder og til videre uddannelse. I forbindelse med individuelt arbejde har eleverne ofte gode forslag til problemer der har udgangspunkt i deres fritidsinteresser eller i relation til andre fag, hvor de selv vurderer et behov for små programmer til fx visualisering eller rutinemæssige udregninger.

Stepwise improvement (fig. 1) er et eksempel på en didaktisk- og metodisk tilgang til arbejdet med it-produktioner. For alle projektforløb gælder at selve processen med fordel kan brydes ned i flere enkeltelementer, i starten med en høj grad af lærerstyrede elevarbejder med gennemprøvede eksempler (vejledninger, tutorials mm) Worked examples (WE).



Figur 1 Stepwise Improvement

Eleverne skal gradvist kunne overtage processen med egen produktion (fig. 2), dels gennem forbedring og løsning af konkrete delopgaver i deres projekt med basis i de gennemprøvede eksempler, dels gennem arbejdet med at udvide kravspecifikationerne til produktet (udvide) og til den færdige produktion (omstrukturere).



Figur 2 Worked examples

Modellen kan bruges som et planlægningsværktøj til hvordan man kommer fra A til B til C, og som sådan er modellen ret lavpraktisk. I stedet for at gå mere eller mindre tilfældigt frem mod et færdigt produktmål, kan eleverne bevæge sig mere systematisk i 3 dimensioner ved dels at forbedre deres eksisterende projekter (f.eks. rette fejl), eller udvide dem (tilføje mere funktionalitet) eller restrukturere (dvs. ændre på strukturen i deres programmer).

I mange forløb vil man vælge at skabe en faglig progression gennem trinvis udvikling af et program. En ulempe ved denne elementorienterede metodik er, at nogle elever ikke formår at samle de mange enkeltelementer til en helhedsorienteret forståelse af det valgte programmeringssprog og udviklingsmiljø, selv om de godt kan gengive og redegøre for de enkelte dele af det udviklede program.
En løsning kan være at bruge kortere selvvalgte projekter til at lade eleverne bearbejde og overføre færdigheder fra obligatoriske forløb. På denne måde opnås en vekselvirkning mellem elevernes syntese af den trinvise præsentation og elevernes analyse af tilsvarende selvvalgte problemstillinger.
De kortere varende projekter egner sig også til brug for individuel evaluering. Man kan desuden kombinere forløbet med tilhørende journal-aflevering.

3.2 Arbejdsformer herunder skriftligt arbejde

Al begyndelse er svær
Tidligt i undervisningsforløbet vil alle elever have gavn af at bearbejde simple eksemplariske opgaver. Disse kan have forskellig fokus, fx på betjening af udviklingsmiljøets værktøjer og grafiske brugerflade, sprogets syntaks, de vigtigste kontrolstrukturer eller strukturering af kildekoden.
Elevernes mulighed for at erkende generelle principper og metoder er generelt større, hvis eleverne bearbejder eksempler og opgaver individuelt, eller i små grupper. Dette letter også muligheden for undervisningsdifferentiering.

Tidsforbruget er ofte stort, når eleverne selv skal bearbejde alle eksempler fra bunden. Der bruges ofte meget tid på mere elementære aspekter som eksempelvis brugerfladedesign, der kan være mere eller mindre adskilt fra fokuspunktet for de stillede opgaver. Er der behov for at øge tempoet i et forløb, eller fastholde eleverne på bestemte pointer i de stillede opgaver, kan man vurdere om man med fordel kan tage udgangspunkt i færdige kodeeksempler.
Der kan være stor afstand mellem at kunne gengive og analysere et eksempel til selvstændigt at omstrukturere, vurdere og overføre til nye sammenhænge. Derfor er bearbejdning af færdige kodeeksempler mere velegnet som supplement frem for som erstatning for individuel bearbejdning. Man kan med fordel benytte mindre indspark i form af ekstra opgaver, kodeeksempler og andet for at sikre alle elever et maksimalt udbytte af tiden. Elever der arbejder hurtigt og koncentreret skal ikke køre i tomgang. Alle indspark kan knytte an i det fælles stofområde, men kan i øvrigt bevæge sig i mange retninger. Der kan både være tale om at differentiere på omfang og dybde, således at progressionen fastholdes.

Logik og systematik
I takt med stigende kompleksitet i de udvalgte problemer, bliver det stadig sværere for eleverne at holde styr på programmets logiske sammenhæng. Som hjælp til at skabe overblik kan man vælge at lade eleverne arbejde med opstilling af storyboards for programmernes virkemåde, eller udarbejde flowdiagrammer for programmets informationsflow, eller for kontrolstrukturerne i udvalgte procedurer. Arbejdet hermed fungerer samtidig som del af dokumentationen for det udførte arbejde. (se endvidere skriftligt arbejde) Strukturering af kildekoden, samt udvælgelse og anvendelse af kontrolstrukturer, kan man træne i mere komplekse eksempler ved at lade eleverne arbejde med at læse, fortolke og omstrukturere færdige programeksempler.
Det er vigtigt, at eleverne øver sig i at kommunikere grundlæggende færdigheder og problemstillinger i det fælles fagsprog.

Bearbejdende undervisningsformer
Man bør supplere de obligatoriske forløb med bearbejdende undervisningsformer, så som caseopgaver eller projekter, for at opnå en bearbejdning af viden i mere komplekse sammenhænge. Særligt i relation til caseopgaverne har læreren høj grad af kontrol med hvilke problemstillinger der vil indgå i arbejdet, og dermed hvilket stof eleverne bearbejder i forløbet.
I projektperioderne vælger eleverne selv deres problemstillinger. Det er dog fornuftigt at udarbejde klart formulerede projektoplæg, som grundigt beskriver et problemfelt, både for at sikre at bestemte elementer kommer til at indgå i projektperioden, men også for at det tydeligt fremgår hvilke mål der prioriteres i den aktuelle projektperiode.
Der skal sikres en gradvis større selvstændighed i arbejdet, således at eleverne er i stand til at arbejde selvstændigt med eksamensprojektet. Derfor bør eleverne inden eksamensprojektet have afprøvet projektarbejdsformen, eksempelvis gennem et forudgående projekt af mindre omfang med god mulighed for feedback og erfaringsudveksling mellem lærer og elev, og eleverne imellem.
Den eksperimentelle programudvikling er en kreativ udviklingsproces, som bedst understøttes gennem dialogen mellem eleven/gruppen af elever og vejlederen eller andre elever om, hvorledes et problem kan gribes an. Projektperioderne er en oplagt mulighed for at trække på ressourcepersonerne blandt eleverne med hensyn til faktuel viden og programmeringstekniske færdigheder.
Elevernes udbytte i projektperioderne er procesorienteret, og især inden for snævre tidsrammer kan elevernes udbytte ikke altid aflæses af deres konkrete produkt, som i nogle tilfælde vil være delvist fungerende programmer. Derfor skal der også i evalueringen lægges vægt på elevens dokumentation af overvejelser, anvendte metoder og redegørelse for valgte løsninger. Den skriftlige dokumentation skal tilpasses de problemstillinger som programmet skal løse. Det er vigtigt at dokumentationen ikke tager overhånd, og fjerner fokus fra - og lysten til at programmere.

I forbindelse med eksamensprojektet bør lærer og elev i fællesskab beslutte, hvorvidt eleven kan anvende et andet sprog og udviklingsmiljø end det primære. Såfremt det ligger uden for lærerens egne kompetencer, må det vurderes om eleven har færdigheder til at klare sig med den vejledning læreren kan give af mere generel karakter. I denne sammenhæng skal man være opmærksom på eventuelle licensproblemer ved brug af anden software end skolens. Det bør tilskyndes, at eleverne planlægger eksamensprojektet således, at det spiller sammen med eksempelvis teknologi- eller teknikfagsprojekter.

Skriftlige arbejder
"Der udarbejdes projektbeskrivelser i form af journaler, herunder et eksamensprojekt. Projektet har et omfang svarende til 20 timers uddannelsestid. Projektet består af et produkt og en journal. Journalen skal beskrive udviklingen af det færdige produkt. Journalen må højst have et omfang af 10 sider.
Projektet udarbejdes inden for rammerne af projektoplæg stillet af skolen. Eksaminanden udarbejder en projektbeskrivelse, der godkendes af skolen, når beskrivelsen er tilstrækkelig fagligt bred og niveaumæssigt relevant.
Sammen med projektet afleverer eksaminanden en journal, der beskriver produktet og dokumentationen heraf. Afleveringstidspunktet skal normalt være senest en uge før eksamensperiodens begyndelse.
Hvis faget har fået tillagt elevtid, skal det skriftlige arbejde tilrettelægges, så der er progression i fagets skriftlighed og sammenhæng til skriftligt arbejde i andre fag i udviklingen af den enkelte elevs skriftlige kompetencer."

(der henvises til uddannelsesbekendtgørelsernes bilag 4: *Elevernes studieforberedende skrivekompetencer*)

Skrivningen har to funktioner med hvert sit formål, tænkeskrivning og formidlingsskrivning. Begge funktioner kan med fordel bringes i anvendelse i arbejdet med projekter i programmering.

Tænkeskrivning: er rettet imod eleven selv, uden at tænke på korrekthed, disposition og læserforventninger.
Eksempler på tænkeskrivnings-genrer i forbindelse med tilrettelæggelse og gennemførelse af produktionsforløb i programmering, som i de fleste tilfælde også kan trænes i udarbejdelsen af journaler:

Idé- og tilrettelæggelsesfase (fremadrettet og åbnende)	Undervejsskrivning (refleksion over igangværende proces)	Evaluerende skrivning (status og refleksion)
Mindmap	Logskrivning	Evaluering af produktion
Brainstorming	Projekt blog	Refleksion over faglig progression
Hv-spørgsmål hvad, hvordan, hvornår, hvorfor, hvilke konsekvenser	Diskussionsfora	Refleksioner over arbejdsprocesser, arbejdsformer og læring, egen indsats i relation til udbytte

Formidlingsskrivning: eleverne skal under anvendelse af programmeringsfaglig viden, grundlæggende metoder i faget og relevant dokumentation kunne give en klar, sammenhængende og nuanceret skriftlig fremstilling i forbindelse med deres produktioner. Den indre censor skal tilkobles så der i rapporten ikke er sprogfejl, genrebrud og andre forstyrrelser i kommunikationen. Her arbejdes bevidst med den sproglige form.

Journalen er en genre som er velegnet til såvel tænkeskrivning som formidlingsskrivning i

forbindelse med mindre projekter og med eksamensprojektet. Dermed vil det skriftlige arbejde også blive elevens egne dokumentation og refleksion over projektets faser og således en værdifuld støtte i forbindelse med den mundtlige eksamen.
Det anbefales at stille et eksempel på en journalopbygning til rådighed for eleverne. En sådan eksemplarisk journal skal ikke omfatte for komplekse programmer, men gerne dække de dele man normalt ønsker at få dokumenteret i en journal så som:

- Forblad
- Kort abstract (censor kan herved orientere sig om opgavens indhold)
- Problemformulering
- Funktionsbeskrivelse (skærmlayout, indtastningsmuligheder, funktionalitet – alt efter hvad det er for et program)
- Dokumentation af selve programmet (overordnet beskrivelse af programmet, detaljeret dokumentation af dele af programmet (flowchart, pseudokode), variabler, objekter, events, igen meget afhængigt af hvad det er for et program)
- Test af programmet
- Konklusion
- Bilag (det er godt at få koden placeret i bilag, da den muligvis ikke ville kunne indeholdes indenfor journalens 10 sider).

Eksamensprojektets skriftlige krav er en journal på højst 10 sider excl. evt bilag

3.3 It
Det er en grundlæggende forudsætning for faget, at skolen it-mæssigt er tilstrækkeligt udstyret både med hensyn til av-udstyr, computere og programmel, således dette kan være til rådighed i undervisningen. Det er også ønskeligt at skolen råder over udstyr der kan anvendes til at eksemplificere kommunikation mellem computere og deres omgivelser i praksis, eksempelvis baseret på programmerbart måleudstyr eller robotter.
Ved brug af Internettet bør det undgås, at eleverne fortaber sig i ustrukturerede søgninger. I faget programmering er det mere rationelt at føre eleverne frem til nogle få steder, der virkelig er værd at konsultere.

3.4 Samspil med andre fag
Der er væsentlig forskel på om faget gennemføres som et valgfag eller som en del af en studieretning.
Hvis faget er en del af en studieretning, sammen med fx Kommunikation/it A, er det oplagt at en del af projekterne kan være tværfaglige forløb med kommunikation og andre fag. Det kan fx være et forløb, hvor man vil præsentere teorien bag det skrå kast i form af en hjemmeside. Det stiller krav til eleverne om at de skal kunne formidle kort og klart (kommunikation), siden skal præsentere sig ordentligt (hvilke farver er det hensigtsmæssigt at bruge – igen kommunikation). Siden skal være interaktiv (programmering). Den teori der danner grundlaget skal være korrekt, og vises på en forståelig måde (fysik). På denne måde opleves også fysik uden for den almindelige fysikundervisning.

Hvis faget er et valgfag er det sværere at opnå samspil med andre fag, da eleverne vil komme fra forskellige studieretninger. Det forhindrer dog ikke, at man kan udnytte tværfagligheden. Det er relevant i programmering at arbejde med algoritmer, fx metoder til iteration, hvor det vil være oplagt at tage emner, som indgår i matematik.

Specielt på 3. år er teknikfagene oplagte til, at eleverne kan benytte teknikfaget som udgangspunkt

for valg af projekter og opgaveløsninger, og lave programmer der understøtter deres faglighed i teknikfaget. Med fornuftig planlægning kan eksamensperioden samstemmes med teknikfagets eksamensperiode. De fleste teknikfag vil kunne anvende elementer fra programmering til at understøtte de produkter de laver, og det vil samtidig give gode helhedsorienterede programmeringsprojekter, selvom nogle elementer i besvarelserne kan komme til at ligge et stykke væk fra programmeringsfagets kernestof.

Elever med teknikfaget *Proces, levnedsmiddel og sundhed* kan eksempelvis bruge programmering til at udvide mulighederne for eksperimentelt arbejde med relation til dataopsamling, procesovervågning, styring og regulering.

I teknikfaget *Design og Produktion – el-retningen*, er det helt oplagt at lave produkter der er programmerbare. Det kan være en mikrocontroller, der programmeres i maskinkode eller et højniveau-sprog af en art (C, Pascal, JAL, Basic eller lignende). Det kan også være programmet til en pc, der kommunikerer med anden elektronik via en af pc-portene.

I biologi og kemi hvor der indgår meget eksperimentelt arbejde, anvendes også dataopsamling. Det udvider området af mulige eksperimenter, at anvende styring og regulering i eksperimenterne fordi eksperimenterne kan gøres uafhængige af undervisningstid. I forbindelse med dataopsamling introduceres ofte nogle 'black boxes' i form af hardware og software til at kontrollere selve dataopsamlingen.

4. Evaluering

4.1 Løbende evaluering

Det anbefales løbende at evaluere med henblik på en fortsat justering af undervisningsforløbets tilrettelæggelse.
Formålet med den løbende interne evaluering er at justere undervisningen og derved optimere elevernes læring. Det anbefales at man efter hvert forløb, emnekreds eller tema ser tilbage på processen og udbyttet heraf. Eleven får herigennem indflydelse på undervisningens tilrettelæggelse, hvilket kan styrke elevaktivitet, motivation og ansvarlighed.
Evalueringen kan tage udgangspunkt i undervisningen (faglighed, pædagogik, engagement, forvaltning af lærerrollen), elevernes indsats (udbytte, arbejdsvaner, forudsætninger mm.), arbejdsformerne, stoffet/emnet, valg af cases (ud af huset aktiviteter, sværhedsgrad, relevans og sammenhæng med anden undervisning) samt arbejdsklimaet i klassen.

Under og ved afslutningen af et undervisningsforløb kan læreren indsamle information om elevernes forståelse af begreberne fx gennem samtale, prøver eller lignende. For at bevare overblikket er det nyttigt at foretage en opsamling af evalueringernes konklusioner.

Det er naturligt at man vurderer elevernes evner til at beherske det valgte programmeringsmiljø, når man hjælper eleverne i den daglige undervisning, både i den første indlæringsfase af sprogets grundelementer, samt i de mere projektorienterede forløb.

Den afsluttende evaluering i faget er baseret på et eksamensprojekt, med et afsluttende produkt og tilhørende journal. Derfor skal eleverne også have træning i at skrive journaler og tilbagemelding om deres standpunkt, og hvordan det kan forbedres.

4.2 Prøveform

Eksamensprojekt og eksamen

Eksaminanden skal inden eksamensudtrækkets offentliggørelse aflevere sit eksamensprojekt i form af produkt og journal, som eksaminator evaluerer som forberedelse til den mundtlige eksamen. Journalen er forinden prøven ikke rettet og kommenteret af læreren.
Eksaminanden afleverer to eksemplarer af journalen. Den ene afleveres til eksaminator; den anden fremsender skolen til censor.
Hvis faget ikke udtrækkes indgår evalueringen af eksamensprojektet i fastsættelsen af årskarakteren.

Skolen skal sikre, at eksaminanden efter afleveringen ikke har mulighed for at ændre i projektet. Hvis projektet afleveres på skolens server, må eksaminanden højst have læserettigheder til det afleverede. Samme regel gælder naturligvis hvis eksterne serverfaciliteter anvendes.
Hvis ikke skolen stiller serverplads til rådighed for eleven, vil det være et fornuftigt krav at produktet også bliver afleveret på diskette eller cd-rom. Eksaminator skal også fx gennem projektoplægget sikre sig tilstrækkelig information så produktet også kan testes. Det kan være database eller serveroplysninger, som forudsætter at produktet kan virke, samtidigt med at kildekoden også er tilgængelig. I de fleste tilfælde vil det være fornuftigt at kræve produktet i maskinlæsbar form.

Afleveringsfristen for eksamensprojektets produkt og journal fastsættes af skolen, dog således at tidsfristen senest en uge før eksamensudtrækningen overholdes.
Produkt og journal skal være til rådighed ved eksaminationen.

Det vil ofte være en fordel at eksaminanden medbringer eget udstyr til at understøtte sin fremlæggelse. Det kan være produktet på en bærbar computer eller en multimediepræsentation..

Efter elevens fremlæggelse af projektet skal der eksamineres med udgangspunkt i projektet. Indholdet af denne eksamination skal være aftalt mellem censor og eksaminator.

(Der henvises i øvrigt til eksamensbekendtgørelsen:
https://www.retsinformation.dk/Forms/R0710.aspx?id=126001)

4.3 Bedømmelseskriterier
Med internettets muligheder for at finde mange gode programmer og programdele, skal det fremgå klart, hvilke dele af programmet eleven selv har fremstillet, og hvornår der er tale om andres programdele og biblioteksmoduler. Oprindelsen skal fremgå af journalen, og eleven skal kunne dokumentere hvordan de bruges.
I tvivlstilfælde kan man bede eleven om at ændre en lille smule på funktionaliteten af programmet. Det kan ligeledes være relevant at bede eleven redegøre for udvalgte kontrolstrukturer i programmet, og argumentere for valget af netop disse frem for andre af eleven kendte kontrolstrukturer.
Det kan også være afklarende at spørge om hvordan programmet er blevet udviklet – hvad har eleven startet med at løse, og hvordan er opbygningen ellers sket. Det kan også give et godt billede af elevens evner, at få oplyst hvilke problemer, der har været i udviklingen af programmet. Især for de dygtige elever viser det noget om deres kompetencer inden for programmering, hvis de kan give forslag til hvordan problemstillingen ellers kunne have været løst.
Journalen indgår kun i evalueringen af elevens præstation i det omfang den bliver inddraget i eksaminationen.

(Der henvises i øvrigt til karakterbekendtgørelsen:
https://www.retsinformation.dk/Forms/R0710.aspx?id=25308)