Instuderingsfrågor inför prov 1 i Programmering 2

Prov 1 omfattar: Kursboken, **kap 1-5**, **sid 7-161**, inkl.:

Övningar: kap 1, sid 22-24
kap 2, sid 78
kap 3, sid 123-125
kap 4, sid 144-146
kap 5, sid 161

Svar till alla frågor kan hittas i kursboken *Programmering 2 med C++*

Alla program- och sidoreferenser hänvisar till kursboken.

Kap 1 Olika programmeringsmiljöer, sid 7-24

1. Vad är den traditionella, procedurala synen på programmering som rådde på 60- och 70-talet?
2. När skedde utvidgningen från C till C++ och vem lade grunden till denna utveckling?
3. Vad är den viktigaste skillnaden mellan C och C++?
4. Varför finns logiska paralleller mellan C/C++ och Unix?
5. Vad var anledningen till att man på 80-talet bytte paradigm inom programmering?
6. Nämn två operativsystem som är programmerade i C.
7. Vad innebär begreppet pekare och vilken relevans har det för programutveckling?
8. Är pekare ett koncept som finns i C eller har man lagt till det senare med C++?
9. Vad innebär det att C är en delmängd av C++?
10. Vad betyder tillägget ++ vid vidareutvecklingen från C till C++?
11. Vad är den historiska orsaken för att C/C++ inte är optimalt för grafiska tillämpningar?
12. För vilka teknologier är C/C++ optimalt och vilka av språkens egenskaper är anledning till det?
13. Vad betyder tillägget **++** vid vidareutvecklingen från C till C++?
14. Är C++ ett standardiserat programmeringsspråk?
15. Ingår biblioteket i C++ standarden?
16. Är Python ett kompilerande eller interpreterande språk?
17. Nämn en fördel av interpreterande språk.
18. Vad har man ersatt måsvingarna { } med i Python?
19. Vilken teknik använder Python för deklaration av variabler?
20. Inom vilket område är Python mest populärt?
21. Är Python ett universellt programmeringsspråk eller ett skriptspråk?
22. I vilka avseenden är Python revolutionerande inom mjukvaruteknologin?
23. Vad innebär det att C++ är ett universellt programmeringsspråk?

24. Är C++ ett interpreterande eller ett kompilerande språk?
25. Är C++ källkod eller maskinkod?
26. Vad är fördelen med case sensitivity i C++?
27. Vilka två steg måste man gå för att se körresultatet av ett C++-program?
28. Beskriv skillnaden mellan kompilerings- och exekveringsfel.
29. Skriver man källkod eller maskinkod när man programmerar?
30. Vilken egenskap borde editorn ha i vilken man skriver programkoden?
31. Vad gör länkningen som ett steg mellan kompilering och exekvering?
32. Redovisa med egna ord de olika typer av fel som kan förekomma vid programmering.
33. Vad bestod den tekniska innovationen av i den datormodell John von Neumann utvecklade 1944?
34. Vilket var det första programmeringsspråk som utvecklades för de första datorerna? Vilka egenskaper hade det? Vad är dess största skillnad till dagens programmeringsspråk?
35. Vad karaktäriserar de programmeringsspråk som kallades för lågnivåspråk? Varför ”låg”?
36. Vilket var det första *högnivåspråket?* Varför *”hög”?*
37. Redogör för skillnaderna mellan begreppen *assemblering, kompilering och interpretering.*
38. Nämn ett exempel på programmeringsspråk som använde en av metoderna i frågan ovan.
39. Vad var det första användningsområdet för programmering?
40. Finns det fortfarande kod som används som är skriven i något av de första programmeringsspråken? Nämn några sådana samt deras användningsområde.
41. Vilket var det första programmeringsspråk som introducerade *kontrollstrukturer* i programmeringen?
42. Vilka är de tre grundläggande kontrollstrukturerna i alla procedurala programmeringsspråk?

Kap 2 Fortsättning med C++, sid 25-78

43. Vad är **`MessageBox()`** och vad gör den?
44. Är **`LPCWSTR`** en operator, en funktion eller en datatyp? Vad gör den i programmet **`MessageBoxVS`** (sid 28) och varför används den där?
45. Är Unicode ett programmeringsspråk? Om ja, är det interpreterande eller kompilerande? Om nej, vad är Unicode då?
46. Vilket behov ledde till uppkomsten av Unicode?
47. Vad är föregångaren till Unicode?
48. Vilken roll spelade tillämpningen av grafik inom IT för uppkomsten av Unicode?
49. Är Unicode en delmängd av ASCII eller omvänt?
50. Ge ett kodexempel på explicit typkonvertering.
51. Hur kan man med escapesekvenser generera i C++ kod de svenska specialtecknen ä, å, ö, Ä, Å, Ö?
52. Vad är radfortsättningstecknet i C++?

53. Ge några exempel på operatorer i C++.
54. Kan följande sats komipleras i C++? **`string name = ”förnamn” + ” ” + ”efternamn”;`**
    Om inte, rätta till koden. Vad kallas **`+`** i satsen ovan?
55. Ge ett exempel på *objektorienterad initiering*.
56. Med vilken enkel datatyp representeras decimaltalskonstanter automatiskt?
57. Med vilken enkel datatyp representeras heltalskonstanter automatiskt?
58. Förklara med egna ord int-regeln vid automatisk typkonvertering.
59. Vad blir b efter satsen **`short b = s1 + s2;`** om man före denna sats har skrivit koden:

```
short s1 = 1;
short s2 = 2;
```

    Testa i ett litet program. Förklara resultatet. Rätta till koden om den är fel.

60. Vad blir **`b`** efter satsen **`long b = s1 + s2;`** om man före denna sats har skrivit koden:

```
short s1 = 1;
short s2 = 2;
```

    Testa i ett litet program. Förklara resultatet. Rätta till koden om den är fel.

61. Följande program kompileringsfel. Rätta till felet.

```
#include <iostream>
using namespace std;
int main()
{
   char letter;
   cout << "Tecknet " << letter << " har koden " << (int) letter;

   letter++;

   cout << "\nTecknet " << letter << " har koden " << letter + 1;
}
```

62. Förutsatt du har rättat kompileringsfelet i fråga 61, varför konverteras **`letter`** till **`int`** i den första **`cout`**-satsen, men inte behöver konverteras i den andra **`cout`**-satsen?
63. Vad händer med datatypen i uttrycket **`letter + 1`** i fråga 61?
64. Varför är **`fib()`** i headerfilen **`Fibonacci.h`** (sid 51) en *rekursiv* funktion?
65. Vad menas med *Beräkningskomplexitet* hos rekursiva funktioner?
66. Kan en **`switch`**-sats ha *tomma case-satser*? Om ja, för vilket ändamål används den?
67. *Algoritmen för platsbyte* kodas på sid 59. Är programmet **`MiniSort`** objektorienterat?
68. Varför misslyckas försöket att modularisera **`MiniSort`**?
69. Beskriv med egna ord parameteröverföringsmetoden *värdeanrop* (*Call by value*).
70. Beskriv med egna ord parameteröverföringsmetoden *referensanrop* (*Call by reference*).
71. Vilka ändringar måste göras i koden, för att lyckas med modulariseringen av **`MiniSort`**?
72. Hur måste en parameter i en metod deklareras för att bli en utparameter?

73. Vad är skillnaden mellan en metods returvärde och metodens utparametrar?
74. Varför (och när) behövs utparametrar, när man kan exportera data från en metod via returvärdet?
75. Hur kan man deklarera en array som parameter i en metod?
76. Var någonstans måste storleken på en array anges när den skickas som parameter till en metod?
77. Vilken parameteröverföringsmetod använder C++ automatiskt på en array som parameter?
78. Ge ett exempel på ett *nästlat anrop* av funktioner.

Kap 3 Klasser, sid 79-125

79. Vad är den traditionella, procedurala synen på programmering som rådde på 60- och 70-talet?
80. Vad är den objektorienterade synen på programmering som kom upp på 80-talet?
81. Mellan vilka två programmeringsspråk går historiskt skiljelinjen mellan procedural och objektorienterad programmering? När ungefär inträffade övergången?
82. Vad var anledningen till paradigmskiftet inom programutveckling?
83. Vilka för- och nackdelar har enligt din åsikt den procedurala synen på programmering?
84. Vilka för- och nackdelar har enligt din åsikt den objektorienterade synen på programmering?
85. Vad menas med *paradigmskifte* i programmeringens historia?
86. Mellan vilka två programmeringsspråk går historiskt skiljelinjen mellan procedural och objektorienterad programmering? När ungefär inträffade övergången?
87. Vad var anledningen till paradigmskiftet inom programutveckling?
88. Vilka för- och nackdelar har enligt din åsikt den *procedurala* synen på programmering?
89. Vilka för- och nackdelar har enligt din åsikt den *objektorienterade* synen på programmering?
90. Är det korrekt att pepparkakor är *klasser* och pepparkaksformen *objekt*?
91. Kan man via *abstraktion* komma från objekt till klass eller är det tvärtom?
92. Om pennor är objekt var kan man hitta klassen *penna*?
93. Av vilka två huvudingredienser består en klass i regel?
94. Anta att *Tal* är en klass. Är *addition()* en metod eller en datamedlem i klassen *Tal*?
95. Anta att *Bil* är en klass. Är *Motor* en metod eller en datamedlem i klassen *Bil*?
96. Vad är skillnaden mellan *funktioner* och *metoder* i C++?
97. Vilka är den objektorienterade programmeringens tre hörnstenar?
98. Vad innebär modularisering på klassnivå?

Kap 4 Logik för blivande programmerare, sid 126-146

99. Varför är ämnet *Logik* relevant för programmering? I vilka delar av ett program används logik?
100. Vad är skillnaden mellan en *instruktion* och ett *villkor?*
101. Är *logiskt uttryck* synonym till *villkor?*

102. Ge några exempel på logiska uttryck. Är temp <= 10 ett logiskt uttryck?
103. Ange koden i C++ som kan lagra värdet till ett logiskt uttryck?
104. Vad skilljer ett sammansatt från ett enkelt logiskt uttryck?
105. Nämn några exempel på logiska operatorer. Är == en logisk operator?
106. Skriv ett sammansatt logiskt uttryck som placerar talet t inom intervallet [1, 15].
107. Skriv ett sammansatt logiskt uttryck som placerar talet t utanför intervallet [1, 15].
108. Bestäm manuellt sanningsvärdena hos de följande logiska uttrycken:

    a) `(8 < 7) && (true || false)`
    b) `!(3 < 3.01) || (!(0==0) && true)`
    c) `(true || !false) && !(!(4*5==1) && false)`

109. Beskriv med egna ord vad en *sanningstabell* är och vad den kan användas till.
110. Döp om i programmet **GuessNEG** (kursboken, sid 137) variabeln **`wrongGuess`** till **`rightGuess`**. Tilldela till den det logiska uttrycket **`guessedNo == secretNo`**, dvs ta bort negationen. Gör de ändringar i programmet som behövs så att programmet fungerar på samma sätt som förut.
111. Formulera som en logisk lag det vardagliga exemplet:

    ”Jag dricker kaffe utan socker OCH utan mjölk.”

    betyder samma sak som

    ”Jag dricker kaffe varken med socker ELLER med mjölk.”.

    Vad kallas lagen i logiken?
112. Hur kan man manuellt bevisa logiska lagar?
113. Har likhetstecknet i *De Morgans lagar* (kursboken, sid 142) samma betydelse som i matematiken? Vad kallas det i logiken?

Kap 5 Filhantering, sid 147-161

114. Vad är motivationen för filhantering i programmering?
115. Vilka två operationer står i centrum för filhantering?
116. Vilket bibliotek behövs för att kunna använda filhanteringsklasser i ett C++ program?
117. Vilken klass ur biblioteket i frågan ovan behövs för att skriva från ett C++ program till filer?
118. När och var någonstans i datorns fil- och mappsystem skapas filen **`Textfil.txt`** med satsen:

```
ofstream fileForWrite("Textfil.txt");
```

119. Vad händer om du exekverar satsen i frågan ovan om filen **`Textfil.txt`** redan finns?
120. Vilken del av satsen i fråga 5 skapar filhanteringsobjektet och vad är referensen till objektet?
121. Skriv kod som behövs för att skriva texten **`"Hallo!"`** till filen som skapas i fråga 108.
122. Skriv kod som stänger filen **`Textfil.txt`**. Vad händer när filen stängs efter användning?
123. Vilken klass behövs för att läsa från filer till ett C++ program?

124. Vad händer i satsen:

```
ifstream fileForRead("Textfil.txt");
```

125. Vad heter datamedlemmen som representerar filslutstecknet i i klassen **`ifstream`**?
126. Av vilken datatyp är filslutstecknet i i klassen **`ifstream`**?
127. Skriv kod som läser allt innehåll från filen som nämns i fråga 124.
128. Modifiera satsen i fråga 124 för att lägga till text till filen **`Textfil.txt`** utan att det gamla innehållet raderas.
129. Modifiera funktionen **`randPasswd()`** (sid 154) genom att implementera en ny lösenord Policy för att generera slumplösenord: 3 gemener, 2 versaler och 2 specialtecken. Inkludera tecknen ? och @ i versalerna, för att de är ”grannar” i ASCII-tabellen.
130. Testa den nya policyn i frågan ovan för att skriva ut de nya slumplösenorden samt tillhörande an vändarnamn till en fil.