Hejsan,

Denna reservdugga går för både kursen TDIU08 och 725G92. Det finns både Ada- och C++-uppgifter på denna dugga men alla uppgifter är inte relevanta för alla studenter. I rubriken för varje uppgift står tydligt vilken laboration som uppgiften är knuten till, vilken uppgift som koden skall skickas in på i studentklienten SC, och vilken kurs som uppgiften är relevant för. När ingen kurskod anges så kan studenter från båda kurserna göra uppgiften. I tabellen nedan finns en översikt:

Kurs	**Relevanta Uppgifter**
725G92	1, 2, 3, 4, 5
TDIU08	1, 2, 3, 4, 6, 7

Psst: TDIU08-studenter, Det finns möjlighet till G på C++-2 i uppgift 6.

Lycka Till!
/Kursledarna

Ada
Laboration 2 [G]

Skickas in som Uppgift/Assignment 1.

Skriv ett program som skriver ut “GOD JUL” på olika “språk”. På t.ex. Temerianska heter god jul “GUD JOL”, på Aedirnska heter det “GAD JEL”, på Kaedwenianska heter det “GID JAL”.

KRAV: Du skall ha ett underprogram som sköter god-jul-utrskriften. Underprogrammet skall ta tre parametrar, två tecken och ett heltal. De två tecknen är de två bokstäver som skall ersätta ’O’ och ’U’ i “GOD JUL”. Heltalet anger hur många utropstecken man vill ha.

KRAV: Huvudprogrammet skall läsa in de två tecknen och heltalet, och skicka dessa som argument till ditt underprogram.

Körexempel 1

```
Mata in två tecken och ett heltal: A E 1
GAD JEL!
```

Körexempel 2

```
Mata in två tecken och ett heltal: U O 3
GUD JOL!!!
```

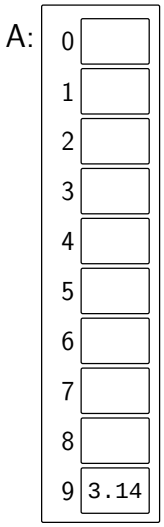
Körexempel 3

```
Mata in två tecken och ett heltal: I A 0
GID JAL
```

Ada
Laboration 3 [G]

Skickas in som Uppgift/Assignment 2.

Skriv ett program som definierar en fältdatatyp som motsvarar bilden nedan (värdet i fältet visar vilken inre datatyp som fältet har). Deklarera sedan en variabel A av denna typ. A skall sedan fyllas med data som användaren får mata in. Programmet skall slutligen skriva ut fältets innehåll (enligt nedanstående körexempel).



Körexempel 1

```
Mata in data: 5.5 4.3 1.0 2.0 3.0 5.678 10.0 3.14 3.14 3.14
Följande data finns i fältet: 5.5 4.3 1.0 2.0 3.0 5.7 10.0 3.1 3.1 3.1
```

Ada
Laboration 4 [G]

Skickas in som Uppgift/Assignment 3.

Givet huvudprogrammet nedan (som även finns på filen test_blood.adb), skapa det paket som behövs (Blood_Handling) d.v.s. en .ads- och en .adb-fil. Paketet skall innehålla datatypen Blood_Type, och underprogrammen *Get* och *Put*. Din datatyp skall vara en post och vara privat. För delvariabeln som representerar *Rh* skall du använda en boolean.

```
1   with Ada.Text_IO;                use Ada.Text_IO;
2   with Blood_Handling;             use Blood_Handling;
3
4   procedure Test_Blood is
5     B  : Blood_Type;   -- En typ för att lagra en blodtyp,
6                        -- har typ (A, B, AB eller 0) och
7                        -- Rh (positiv/negativ, d.v.s. sant/falskt)
8   begin
9     Put("Mata in en blodtyp: ");
10    Get(B);    -- Inmatningsformat t.ex. "AB+", "0-", "A-", o.s.v.
11    Put("Du matade in ");
12    Put(B); -- Utmatningsformat: t.ex. "AB Positiv", "B Negativ"
13    New_Line;
14  end Test_Blood;
```

TIPS: Inmatningen är alltid exakt 2 eller exakt 3 tecken. Endast om typen är ”AB” behöver ett tredje tecken läsas.

Körexempel 1

```
Mata in en blodtyp: AB-
Du matade in AB Negativ
```

Körexempel 2

```
Mata in en blodtyp: 0+
Du matade in 0 Positiv
```

Körexempel 3

```
Mata in en blodtyp: A-
Du matade in A Negativ
```

Ada
Laboration 5 [G]

Skickas in som Uppgift/Assignment 4.

Skriv ett program som låter användaren mata in ett heltal. Programmet skall sedan läsa in så många nya flyttal och beräkna produkten av dessa. Resultatet skall skrivas ut på skärmen. Du kan anta att användaren alltid matar in så många flyttal som heltalet anger.

KRAV: Lös med rekursion. Inga loopar är tillåtna.

Körexempel 1

```
Mata in ett heltal: 1
Mata in 1 flyttal: 5.0
Produkten blev 5.00
```

Körexempel 2

```
Mata in ett heltal: 3
Mata in 3 flyttal: 4.5 5.1 6.9
Produkten blev 158.35
```

Körexempel 3

```
Mata in ett heltal: 10
Mata in 10 flyttal: 8.8 3.3 2.2 7.7 4.4 2.2 0.6 0.4 0.3 10.0
Produkten blev 3428.61
```

Ada (Endast 725G92)

Laboration 6 [G]

Skickas in som Uppgift/Assignment 5.

På filen FALSE_TIMES.TXT ligger det klockslag lagrade (ett per rad). Formatet på dessa klockslag skall vara ”TT.MM” (d.v.s alltid 5 tecken). Det har dock smugit sig in falska klockslag i filen, falska på så sätt att timmen överstiger 23 eller minutrarna överstiger 59. Skriv ett program som öppnar filen, läser igenom den och skriver ut alla falska klockslag. Följande är ett exempel på hur filen kan se ut:

```
12.43
19.93
08.26
09.59
04.64
16.00
14.45
10.30
24.56
03.32
13.37
64.96
```

OBS! Filen har godtyckligt många rader.

KRAV: Programmet skall INTE hantera/fånga undantag (eng. *exceptions*) och programmet får inte avslutas med END_ERROR.

Körexempel 1

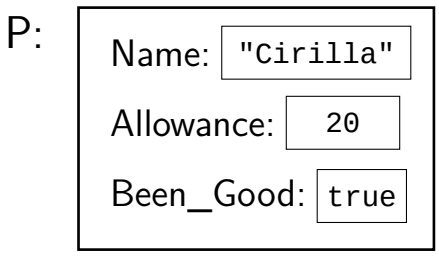
```
Följande falska klockslag hittades:
19.93
04.64
24.56
64.96
```

C++ (Endast TDIU08)

Laboration 3 [G]

Skickas in som Uppgift/Assignment 6.

Definiera en ny typ enligt figuren nedan. Din typ skall vara en *struct* och innehålla de delar som figuren visar. Datat i varje delvariabel visar vilken typ som den delvariabeln skall vara.



Skapa sedan ett huvudprogram som deklararerar en variabel P av din nya typ. Programmet skall sedan göra följande:

1. Låta användaren mata in data som lagras i P.
2. Deklarera en variabel av typen *vector*. Vektorns inre typ skall vara den struct-typ som du definierat.
3. Stoppa in P på första plats i vektorn.
4. Slutligen skall programmet skriva ut datat som ligger lagrat i vektorns första element **(du skall inte använda variabeln P i detta steg)**.

Körexempel 1

```
Mata in namn: Cirilla
Mata in veckopeng: 20
Mata in snäll/stygg: snäll

Cirilla är ett snällt barn med 20 kr i veckopeng
```

OBS: Om man matar in ”snäll” motsvarar detta alltså att Been_Good sätts till `true`. ”stygg” skall alltså ge värdet `false` på Been_Good.

EXTRA FÖR LAB 2: Om du gör två underprogram *get* och *put* och använder dessa i steg 1 resp. steg 4 ovan kan du även få G på C++-laboration 2 (givet att underprogrammen gör vettiga saker och parametrarna är rätt med avseende på referenser och const). Bägge underprogrammen skall endast ha en parameter.

C++ **(Endast TDIU08)**

Laboration 4 [G]

Skickas in som Uppgift/Assignment 7.

På filen FALSE_TIMES.TXT ligger det klockslag lagrade (ett per rad). Formatet på dessa klockslag skall vara ”TT.MM” (d.v.s alltid 5 tecken). Det har dock smugit sig in falska klockslag i filen, falska på så sätt att timmen överstiger 23 eller minutrarna överstiger 59. Skriv ett program som öppnar filen, läser igenom den och skriver ut varje falskt klockslag. Följande är ett exempel på hur filen kan se ut:

```
12.43
19.93
08.26
09.59
04.64
16.00
14.45
10.30
24.56
03.32
13.37
64.96
```

OBS! Filen har godtyckligt många rader.

KRAV: Programmet skall INTE hantera/fånga undantag (eng. *exceptions*) och programmet får inte avslutas med END_ERROR.

Körexempel 1

```
Följande falska klockslag hittades:
19.93
04.64
24.56
64.96
```