IN1020 Obligatorisk Innlevering 1
Digital Representasjon og Programmering

29. august 2024

1 Introduksjon

I denne obligen skal vi se nærmere på digital representasjon og hvordan virkeligheten kan representeres digitalt i en datamaskin. Vi skal også se på programmering i LMC. **Sørg for at du har lest og forstått alt før du begynner på oppgaven.**

2 PLU Koder

Du jobber som godt betalt assembler-konsulent for LMC Daligvare AS i 1987. På denne tiden brukte matbutikker det man kalte PLU (Price-LookUp) koder. Dette er en en tresifret kode som kunne tastes inn i et kassa-apparat (se Figur 1) for å skanne varer som ikke hadde strekkode. Eksempler er frukt og grønnsaker uten emballasje. Kassereren slo inn slike koder når hen skannet varene fra båndet, gjerne sammen med et antall, og kassa-apparatet regnet ut totalsummen av alle varene (se Tabell 1).

Figur 1: Et gammeldags kassa-apparat.

Hva	PLU	Pris (kroner)	Antall	Totalsum
Eple	405	7	1	7
Rundstykke	307	4	2	15
Appelsin	303	10	1	25

Tabell 1: Kassereren kan slå inn PLU koder og antall, og kassen holder styr på totalsummen.

3 Oppgaven

Din oppgave som assemblerutvikler er å lage et moderne kassa-apparat. Du innser at systemet kan optimaliseres ved å bruke LMC for å løse oppgaven. 3-sifrede PLU koder passer nemlig godt for LMC, som representerer verdier i titallsystemet med tre siffere. I tillegg må hver vare ha sin egen pris (per stykk). Denne dataen må lagres i minnet til LMC. Av erfaring vet du at det kan være smart å tenke igjennom andre egenskaper matvarer kan ha, som kan være nyttig å representere digitalt. Dette kan spare deg for tid senere. Det er snart helg og du vil ikke jobbe overtid.

3.1 Et register med PLU koder

- Kom på fire ekstra egenskaper matvarer kan ha som det kan være nyttig å representere digitalt. Her er det ingen fasit, men opp til deg å reflektere over hvilke egenskaper som kan være nyttige å representere digitalt.
  - PLU
  - Pris
  - ?
  - ?
  - ?
  - ?
- Gi en kort **beskrivelse** av hva disse kan bli brukt til (en setning maks). Lag en tabell som vist under, hvor du også angir hvor mange **celler** du trenger, hvilke **verdier** som skal kunne representeres, og om det som ligger i cellen er et heltall eller tekst (**datatype** i tabellen under).

Hva	Beskrivelse	Antall celler	Verdier	Datatype
PLU	Koden kassereren slår opp	1	0 til 999	Heltall
Pris	Prisen til varen, brukt til å beregne totalpris	??	??	??
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-

3.2 Minnestruktur og minnebruk

LMC har et relativt sparsommelig minne på 100 adresserbare celler, nummerert fra 0 til 99 som vist under. Hver celle kan holde et heltall (desimaltall) mellom -999 og +999[1]. Disse brukes til både kode og data:

0	1	2	3	4	5	6	7	8	9
10	11	12	13	14	15	16	17	18	19
20	21	22	23	24	25	26	27	28	29
30	31	32	33	34	35	36	37	38	39
40	41	42	43	44	45	46	47	48	49
50	51	52	53	54	55	56	57	58	59
60	61	62	63	64	65	66	67	68	69
70	71	72	73	74	75	76	77	78	79
80	81	82	83	84	85	86	87	88	89
90	91	92	93	94	95	96	97	98	99

- Ta for deg egenskapene ved matvarer du kunne tenke deg å representere digitalt fra Seksjon 3.1, som PLU, stykkpris og de fire egenskapene du har funnet på selv. Hvor mange minneceller vil du trenge for å representere èn vare? Lag en tabell som vist under.

Celle	Hva
0	PLU
1	Pris
2	?
3	?
4	?
..	?

- Dersom *alle* minnecellene til LMC skulle brukes til å representere matvarer, hvor mange matvarer har du da plass til?

- Hvor ser du for deg at dataene for disse skal lagres i maskinen, samt koden din? Ta utgangspunkt i tabellen over og skissér. Du kan for eksempel bruke blå farge på cellene med registeret ditt, og rød farge på cellene som inneholder koden din, samt dataene koden jobber med. Her er det ikke noe fasitsvar, men du bør tenke litt på hvordan du skal få lagret dataene dine i maskinen før koden kjører.

3.3 Implementasjon

Du skal nå implementere et enkelt kassa-apparat i LMC som man kan bruke til å slå inn varer og beregne totalpris. Programmet skal fungere slik:

1. Programmet leser inn en PLU kode.

[1] En ekte datamaskin er så klart binær.

- Du står fritt til å velge at kassereren aldri taster feil kode, og trenger dermed ikke ta hensyn til dette.
- Dersom PLU koden er 0, skal programmet skrive ut totalsum for varene og avslutte.

2. Et antall varer fra 1 og oppover.
3. Programmet ganger prisen på varen med antall og oppdaterer totalsum.
4. Programmet fortsetter fra 1).

Programmet skal støtte minst **to** PLU koder (du står fritt til å ta med flere). Disse, og prisene for hver PLU, må forhåndslagres i maskinen med bruk av pseudoinstruksjonen **DAT**. Hver matvare representeres altså slik:

- En PLU kode i èn celle.
- En pris i èn celle.

Du står fritt og er velkommen til å legge til og bruke flere egenskaper som du har funnet på selv, men det kreves ikke for å bestå obligen.

Konsultentselskapet ditt krever også at du legger inn ditt IFI brukernavn helt i slutten av programmet ditt med DAT pseudoinstruksjoner.

4 Tips

- Begynn oppgaven uten å ta hensyn til antall varer (punkt 2 i listen over).
- Begynn med èn PLU kode og pris og få programmet til å fungere før du legger til flere varer.
- Her er det lett å "miste"arbeidet sitt fordi man begynner på noe nytt og havner i trøbbel. Lagre derfor unna assemblerkoden hver gang du får til noe som virker.
- Du kan legge til flere egenskaper som du har funnet på selv og anvende dem i programmet ditt, men det er ikke nødvendig for å få godkjent.
- Her går det an å bruke selvmodifiserende kode (avansert) eller en del etiketter (enklere, men mer kode) for å sjekke hvilken PLU bruker har tastet inn. Tilpass oppgaven etter dine egne ambisjoner.
- Du vil trenge en løkke for å gange pris med antall, og oppdatere en totalsum.

5 Innlevering

Leveringsfrist: 17. September 2024 klokken 23:59. Følgende tre filer skal leveres på Devilry:

1. **oppgaver.pdf**
   - Besvarelse på oppgavene (Seksjon 3.1 og 3.2) i PDF-format.
2. **kode.txt**
   - Assemblerkoden for implementasjonen din (Seksjon 3.3).
   - Koden skal kunne kopieres inn i LMC og kjøres uten modifikasjoner i henhold til kravene i Seksjon 3.3.
   - Koden skal kjøre i Peter Higginson sin simulator, som dere finner her: `https://peterhigginson.co.uk/lmc`
3. **readme.txt**
   - Kort forklaring av hvilke PLU-koder programmet ditt bruker. Hvis du har utfordringer med implementasjonen, beskriv:
     - Hvilke utfordringer du har møtt
     - Hvor langt du har kommet

Viktig: Hvis du ikke klarer å løse oppgaven helt komplett, er det viktig at du viser til et ærlig forsøk på å løse oppgaven etter beste evne. Jo bedre du demonstrerer din tankegang rundt oppgaven, desto bedre vil gruppelærer ha mulighet til å gi deg gode tilbakemeldinger, som igjen vil bidra til at du får en bedre forståelse for faget.

Lykke til!