Inlämningsuppgift 3 – Introduktion till C-programmering
DA105A – Programmering med C: Grundkurs, 7,5 hp

Innehåll

1 Inledning

Denna inlämningsuppgift syftar till att ge färdigheter i användandet av arrayer och pekare. Uppgiften visar på det intima släktskapet mellan array och pekare genom att ni i deluppgift 1 ska utföra ett antal operationer på en array och sedan i deluppgift 2 ska samma operationer utföras men med pekararitmetik.

- *C How to Program – 5th Edition av H.M. Deitel och P.J. Deitel,*
  eller
- *C How to Program – 6th Edition av H.M. Deitel och P.J. Deitel.*

Kursboken kommer refereras till med beteckningen [DEITEL].

1.1 Redovisning

Laborationen skall redovisas genom att följande material produceras, och levereras via It's learning [1].

- En kort rapport, i pdf format, som beskriver de erfarenheter du gjort under laborationen.
- En eller flera källkodsfiler, med programmet för er implementation av uppgiften.

All kod ni skickar in ska följa de instruktioner som beskrivs i dokumentet ”coding_style.pdf”, denna hittar ni på kurshemsidan på It's learning[1].
Observera att kod som inte följer dessa instruktioner kommer att skickas tillbaka även applikationen exekverar korrekt.

Inlämningsuppgift 3 ska skickas in i en zip-fil via It's learning.

1.2 Förberedelser

Följande förberedelser rekommenderas:

- Läs igenom föreläsningsanteckning 6 och 7.
- Läs igenom kapitel 6 och 7 i kursboken [DEITEL]. Notera att detta gäller både för 5th och 6th Edition av kursboken.

2 Inlämningsuppgiften

Uppgiften består av två delar som bägge lämpligen implementeras i en källkodsfil.
Mot slutet av detta dokument finner ni ett kodskelett som ni ska utgå ifrån. Börja med att skapa en c-fil och kopiera in detta kodskelett. I den givna koden finns sex funktionsprototyper, deluppgift 1 går ut på att ni ska implementera funktionerna för de tre första funktionsprototyperna och i deluppgift 2 ska ni implementera funktionerna för de tre sista.
Det är inte tillåtet att ändra på de givna funktionsprototyperna eller på koden i main-funktionen. Det går dock bra att lägga till fler funktioner vid behov.

2.1 Deluppgift 1 – Arrayer

Denna deluppgift går ut på att ni utifrån ett givet kodskelett ska implementera tre funktioner. En funktion som fyller en array med slumpmässigt genererade heltal, en funktion som sorterar arrayens element i nummerordning samt en funktion som skriver ut hela arrayen och en sträng text. En mer detaljerad beskrivning av dessa funktioner finner ni längre ner i dokumentet. Begrunda först följande exempel.

2.1.1 Arrayer - exempel

I språket C är det programmerares ansvar att hålla ordning på en arrays storlek. Nedan följer ett exempel på detta.

Vi deklarerar en integer array med 10 element. Sedan tilldelar vi det första elementet på position 0 värdet 1 och det sista elementet på position 9 tilldelar vi värdet 10.

```
int array[10];

array[0] = 1;
array[9] = 10;
```

Vad händer om vi tilldelar denna array ett värde på position 10? Detta skulle i Java ge följande felmeddelande ”ArrayIndexOutOfBoundsException”, men i C händer följande.

```
array[10] = 11;
```

Samma sak gäller för negativa indexvärden.

```
array[-2] = 12;
```

Detta leder till möjligheten att man skriver över bitar som används av andra variabler som är deklarerade före och efter arrayen.
Det är således mycket viktigt att hålla ordning på storleken av en array.

2.1.2 Implementation de tre första funktionerna (version 1)

Din uppgift är att implementera funktionen för funktionsprototypen nedan.

```
void fill_rand_ver1(int array[], int size);
```

Funktionen tar en integer-array och en integer som argument. Iterera nu igenom arrayen och tilldela varje element ett slumpmässigt heltal i intervallet 1-10. Använd biblioteksfunktionen rand() i kombinations med modulus för att åstadkomma detta.

Implementera nu funktionen för nästa funktionsprototyp.

```
void sort_ver1(int array[], int size);
```

Även denna funktion tar en integer-array och en integer som argument.
Använd sorteringsalgoritmen selectionsort som beskrivs nedan till att sortera arrayens element.

Algoritm Selectionsort

Nedan ges delar av koden för sorteringsalgoritmen Selectionsort.

```
void sort_ver1(int array[], int size)
{
        int i, j, temp;
        for(i = 0; i < size; i++){
                for(j = i + 1; j < size; j++){
                        if(array[j] < array[i]){
                                /* Gör en ”swap”, dvs byt plats på elementen
                                   på position i och j.
                                    Använd hjälpvariabeln temp. */
                        }
                }
        }
}
```

Selectionsort använder här en nästlad for-loop till att iterera igenom och jämföra arrayens alla element. Om värdet på ett element med lägre index är större än värdet på ett element med högre index utförs en så kallad ”swap”, d v s två elements värden byter plats.

Till detta behövs en temporär variabel för att hålla värdet för det främre elementet så att detta inte skrivs över. Den yttre for-loopen itererar igenom alla elementen i arrayen medan den inre for-loopen börjar från elementet efter positionen för den yttre loopen. Sedan itererar den inre loopen igenom den återstående delen av arrayen och jämför varje element med värdet på elementet där den yttre loopen befinner sig.

Implementera nu funktionen för den tredje funktionsprototypen.

```
void print_values_ver1(char string[], int array[], int size);
```

Använd biblioteksfunktionen printf till att skriva ut character-arrayen som tas som argument. Funktionen printf behöver inte veta längden på character-arrayen, utan skriver ut alla tecken tills den påträffar ett null elementet '\0'. En character-array ska alltid avslutas med ett null element, det måste således finnas ett extra element över till detta. Använd ”%s” i formatsträngen till att precisera att det är en sträng som ska skrivas ut. Skriv slutligen ut arrayens alla värden med hjälp av printf och en for-loop.

2.2 Deluppgift 2 – Pekare

2.2.1 Implementation de tre sista funktionerna (version 2)

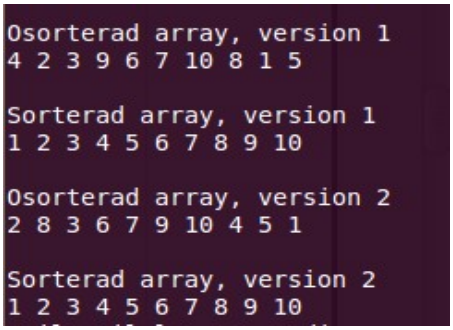
I denna deluppgift ska ni implementera tre funktioner för de tre sista funktionsprototyperna som ska göra exakt samma sak som de tre funktionerna som ni just har skrivit. Denna gång ska ni däremot använda pekararitmetik (se sektion 7.8 [DEITEL] om pekararitmetik). Syftet är att ni ska öva på att använda pekare, det är alltså **inte** tillåtet att använda index för att utföra operationerna på arrayen.

Arrayer och pekare är mycket nära besläktade, ett arraynamn kan ses som en konstant pekare till första elementet i en array. Arraynamnet array motsvarar alltså &array[0], d v s adressen på det första elementet i arrayen.
Nedan följer ett exempel på hur man kan skriva ut alla värden i en array genom att använda pekararitmetik.

```
void print_array(int *pointer, int size)
{
        int *pi;
        for(pi = pointer; pi < (pointer + size); pi++)
                printf("%d ", *pi);
}
```

Pekaren pi initieras till att peka på första elementet i en array. Så länge värdet för pekaren pi är mindre än arrayens storlek ökas värdet på pi till att peka på nästa element i arrayen.
Skriv ut värdet som pi pekar på genom att använda *-operatorn.

Utskriften av en körning av det slutliga programmet ser ut som nedan.




```
//----------------------- Kodskelett start ----------------------

#include <stdio.h>

#define SIZE 10 /* Deklarerar en symbolisk konstant */

/* Funktionsprototyper */
void fill_rand_ver1(int array[], int size);
void sort_ver1(int array[], int size);
void print_values_ver1(char string[], int array[], int size);

void fill_rand_ver2(int *pointer, int size);
void sort_ver2(int *pointer, int size);
void print_values_ver2(char *string, int *pointer, int size);

int main()
{
      int array[SIZE]; /* Deklarerar en array av integers */

      srand(time(0)); /* Frö till funktionen rand()*/

      fill_rand_ver1(array, SIZE); /* Fyll arrayen med slumptal */

      /* Skriv ut textsträngen och arrayens värden */
      print_values_ver1("Osorterad array, version 1", array, SIZE);

      sort_ver1(array, SIZE); /* Sortera arrayen */

      /* Skriv ut textsträngen och arrayens nu sorterade värden */
      print_values_ver1("Sorterad array, version 1", array, SIZE);

      int *pointer; /* Deklarerar en pekare av typen integer */

      pointer = array; /* Pekaren initieras att peka på
                          första elementet i arrayen */
```

```
	fill_rand_ver2(pointer, SIZE); /* Fyll arrayen med nya slumptal */

	/* Skriv ut textsträngen och arrayens värden */
	print_values_ver2("Osorterad array, version 2", pointer, SIZE);

	sort_ver2(pointer, SIZE); /* Sortera arrayen */

	/* Skriv ut textsträngen och arrayens nu sorterade värden */
	print_values_ver2("Sorterad array, version 2", pointer, SIZE);

	return 0;
}

//----------------------- Kodskelett slut ----------------------
```

Referenser

[1] It's learning
http://www.itslearning.com/elogin