Tentamen - Programmering

DVA117

----------FACIT----------

Akademin för innovation, design och teknik

Torsdag 2017-11-02

An English translation of the entire exam follows after the questions in Swedish

Skrivtid: 08.10 – 11.30
Hjälpmedel: Valfritt icke-elektroniskt material
Lärare: Caroline Uppsäll, 0704616110
(kan nås på telefon om du frågar tentavakten)

Preliminära betygsgränser

Betyg 3:	11p
Betyg 4:	16p
Betyg 5:	19p
Max:	**22p**

Allmänt

- All kod skall skrivas i standard ANSI C.
- Skriv tydligt vilken uppgift/deluppgift ditt svar anser.
- Skriv endast på bladets ena sida.
- Referera inte mellan olika svar.
- Om du är osäker på vad som avses i någon fråga, skriv då vad du gör för antagande.
- *Oläsliga/oförståeliga/ostrukturerade svar rättas inte.*
- Kommentera din kod!
- Det är inte tillåtet att använda goto-satser och globala variabler
- Tips: Läs igenom hela tentamen innan du börjar skriva för att veta hur du ska disponera din tid.

----------FACIT----------

Lycka till!
/Caroline

Uppgift 1 [1p]

Vilket av följande påståenden (A, B, C, D eller E) är sant?

A. If-satsen används till iteration, switch-satsen till selektion
B. If-satsen används till selektion, switch-satsen till iteration
C. If-satsen gör val baserat på villkor, switch-satsen gör val baserat på värde
D. If-satsen gör val baserat på värde, switch-satsen gör val baserat på villkor
E. Inget av påståendena ovan stämmer.

Med If-sats menas strukturen if - else if - else och med switch-sats menas strukturen switch-case

Uppgift 2 [2p]

Nedan ser du tre olika program innehållande loopar. Två av programmen ger likadana utskrifter, vilket program ger inte samma utskrift som de andra två.

A

```c
#include <stdio.h>

int main(void)
{
    int i = 1, sum = 0;
    while(sum <= 20)
    {
        sum = sum + i;
        if(sum%2 == 0)
            i = sum + 1;
        else
            i++;
        printf("%d   ", sum);
    }
    return 0;
}
```

B

```c
#include <stdio.h>

int main(void)
{
    int i = 1, sum = 0;
    do{
        sum = sum + i;
        if(sum%2 == 0)
            i = sum + 1;
        else
            i++;
        printf("%d   ", sum);
    }while(sum <= 20);
    return 0;
}
```

C

```c
#include <stdio.h>

int main(void)
{
    int i, sum;
    for(i = 1, sum = 0; sum <= 20; sum = sum + i)
    {
        if(sum%2 == 0)
            i = sum + 1;
        else
            i++;
         printf("%d   ", sum);
    }
    return 0;
}
```

RÄTT SVAR: C

I for-loopen uppdateras summa i slutet av loopen, alltså efter utskriften medan summan i de övriga uppdateras i början, innan utskriften

Uppgift 3 [4p]

För följande funktionsanrop, ange funktionshuvudet (dvs det som används när man definierar och deklarerar funktionen).

Antag följande egendefinierade typ

```
struct date{
    int year,month,day;
};
```

Utgå från att följande deklarationer finns i programmet.

```
struct date toDay, *anyDay;
int y, *u;
float arr[10], z;
```

Exempel:

Funktionsanrop:	Ange deklarationen för
`y = func(6.3);`	`func()` svar: `int func(float x);`

Förklaring: Eftersom func() tar ett flyttal som argument måste parametern vara av typen float (vad den heter är mindre viktigt).
Returvärdet tas emot i y som är deklarerad som en int, därför måste func() returnera en int.

Uppgift:

Funktionsanrop:	Ange deklarationen för
1) `print(anyDay, "some date", arr);`	`print()`
`void print(struct date *d, char *str, float arr[]);`	
`Ok med char str[], char str[10] eller void *str`	
`Ok med float *arr`	
2) `z = foo(&u, arr[0]);`	`foo()`
`float foo(int **x, float y);`	
3) `toDay.month = compute(&y, *anyDay, 8);`	`compute()`
`int compute(int *y, struct date d, int x);`	
4) `toDay = start();`	`start()`
`struct date start(void);`	

Uppgift 4 [7p]

Utgå från följande kod:

```
#include <stdio.h>
#include <stdlib.h>

typedef struct
{
    char weekday[10];
    int year, month, day;
}date;

void addData(date *days, int numberOfDays);
void writeToFile(char *fileName, date toDay[], int numberOfDays);

int main(void)
{
    date days[7];
    addData(days, 7);
    writeToFile("file.txt", days, 7);
    return 0;
}

void addData(date *days, int numberOfDays)
{
    //Deluppgift 4.1
}
void writeToFile(char *fileName, date toDay[], int numberOfDays)
{
    FILE *fp;
    int i;
    fp = fopen(fileName, "w");
    if(fp != NULL)
    {
        fprintf(fp, "%d\n", numberOfDays);
        for(i = 0; i < numberOfDays; i++)
            fprintf(fp, "%s\n%d %d %d\n", (toDay[i]).weekday, (toDay[i]).year,
                                          (toDay[i]).month, (toDay[i]).day);
    }
    fclose(fp);
}
```

Deluppgift 4.1 (2p)

Implementera funktionen `addData`, du ska använda den funktionsdeklaration som finns i programmet ovan och får inte ändra någonting i det befintliga programmets implementation. `addData` ska fylla arrayen `days` med information om veckodagar, hela arrayen ska fyllas. Inmatningen ska ges från användaren. Du ska skriva implementationen på ett sådant sätt att det för användaren är enkelt att förstå vilken information som ska matas in.

Lösningsförslag

```
void addData(date *days, int numberOfDays)
{
    int i;
    for(i = 0; i < numberOfDays; i++) (1p)
    {
        printf("day: ");
        gets((days+i)->weekday);
                                        (1p)
        printf("year, month, day: ");
        scanf("%d %d %d", &(days+i)->year, &(days+i)->month, &(days+i)->day);
        while(getchar()!= '\n');
    }
}
```

Deluppgift 4.2 (3p)
Funktionen `writeToFile` sparar innehållet i arrayen `days` till textfilen `file.txt`.
Skapa en ny funktion som sparar **samma** information som `writeToFile` men till en binär fil. Funktionen ska heta `writeToBinaryFile`. Du ska skriva både funktionshuvud och implementation samt ange hur anropet till funktionen skulle se ut (från main).

Som hjälp får du funktionsdeklarationen för fwrite( )
`size_t fwrite(const void *ptr, size_t size, size_t count, FILE *stream);`
`size_t` är en unsigned int

Lösningsförslag

```
void writeToBinaryFile(char *fileName, date toDay[], int numberOfDays) //date *toDay (0.5p)
{
    FILE *fp;
    fp = fopen(fileName, "w"); (1p)
    if(fp != NULL)
    {
        fwrite(&numberOfDays, sizeof(int), 1, fp);  (1p)
        fwrite(toDay, sizeof(date), numberOfDays, fp);
    }
}

Anrop: writeToBinaryFile("fil", days, 7); (0.5p)
```

Deluppgift 4.3 (2p)
`days` i programmet ovan är en array av typen `date`. Antag att programmet deklarerade (i main) en pekare till `date` istället för en array av typen (se nedan). Skapa en ny funktion som dynamiskt allokerar minne för ett antal datum (`date`) i minnet. Antalet ska tas som parameter till funktionen. Du ska skriva både funktionshuvud och implementation samt ange hur anropet till funktionen ser ut (från main). Man ska kunna arbeta med det dynamiskt allokerade minnet även utanför funktionen.

```
~~date days[7];~~
date *days = NULL;
```

Lösningsförslag

```
date *allocateMemory(int numberOfDays)
{
    date *days = (date *)malloc(sizeof(date)*numberOfDays);
    //date days = (date *)calloc(sizeof(date), numberOfDays);
    return days;
}
```

ELLER

```
void allocateMemory(date **days, int numberOfDays) (0.5p)
{
    *days = (date *)malloc(sizeof(date)*numberOfDays); (1p)
    //*days = (date *)calloc(sizeof(date), numberOfDays);
}

Anrop:  days = allocateMemory(7); (0.5p)

        ELLER

        allocateMemory(&days, 7);
```

Uppgift 5 [1p]

Beskriv med en mening vad en pekare är

Rätt svar

En pekare är en variabel (pekarvariabel) som innehåller en adress (pekarvärde) till någonstans i minnet.

Uppgift 6 [4p]

Antag följande inkluderingar och deklarationer,

```
#include <string.h>
#include <stdlib.h>
#include <time.h>

int func(char *str, int x, int *y);

int random, x = 4, y = 9, result;
float arr[10];
char str1[20], *str2, z = '\0';
srand(time(0));
```

Skriv av nedanstående punkter på ditt svarspapper och markera därpå vilka som inte är tillåtna/korrekta satser, motivera också varför de inte är tillåtna/korrekta.
För poäng krävs både att rätt satser markeras och att en korrekt motivering ges.

```
A. random = rand()%100 //Semikolon (;) saknas  (0.5p)
B. x + y = result; //Uttrycket (beräkningen måste vara till
                     höger om '=' (0.5p)
C. y+=result; //resultat är visserligen inte initierad men satsen går ändå att
   utföra – med skräpvärde
D. str1 = "programmering"; //Arrayer måste tilldelas element för
                              element eller så måste strcpy användas (0.5p)
E. strcpy(str2, "med C"); //str2 pekar ingenstans (0.5p extra)
F. str1[10] = z;
G. arr[x] = arr[++y]; //++y ökar y med 1 innan den används, man
                        försöker accessa index 10 i en array med 10
                        platser (index 0 till 9) – index out of bounds
                        (0.5p)
H. int *p = &x;
I. result = func(&str2, y, *x); //Första parameter är char*, str2 är en
                                  char* – & behövs inte. Sista
                                  parameter är en heltalspekare, x är
                                  ett heltal – vi måste skicka adressen
                                  där x ligger i minnet (&x) (1p)
J. result = strlen(str1);
K. puts(str2); //stdio.h är inte inkluderad, str2 pekar ingenstans (0.5p)
L. str1 = str2; //arrayer (str1) går inte att tilldelas (inte en
                  pekarvariabel)  (0.5p)
```

Uppgift 7 [3p]

Beskriv med varsin mening de tre steg som genomförs då man bygger sitt projekt

- Preprocessorn
- Kompilering/kompilatorn
- Länkning/länkaren

Rätt svar

Preprocessor är det första steget där alla # tas om hand (bibliotek inkluderas och makron expanderas). (1p)

Kompilatorn kontrollerar så att koden är syntaxtisk korrekt samt utför oprimeringar på koden, en o-fil för varje c-fil genereras. (1p)

Länkningen innebär att alla o-filer samt bilbioteksfiler länkas samman till en körbar .exe-fil. (1p)

Exam - Programming

DVA117

----------SOLUTIONS----------

School of Innovation, design and technology

Thursday 2017-11-02

Writing time: 08.10 – 11.30
Aids: Any non-electronic material
Examiner: Caroline Uppsäll, 0704616110
(Can be reached by telephone if you ask the exam guard)

Preliminary grading
Grade 3: 11p
Grade 4: 16p
Grade 5: 19p
Max: **22p**

Generally

- All code should be written in standard ANSI C.
- Write clearly what task/sup-task your answers consider.
- Do only use one side of the paper.
- Do not refer between answers.
- If you are unsure of a meaning of a question, write down your assumption.
- *Unreadable/incomprehensible answers will not be marked.*
- Comment your code!
- It is not allowed to use goto-statements or global variables
- Hint: To know how to allocate your time, read through the entire exam before you start writing.

----------SOLUTIONS----------

Good luck!
/Caroline

Question 1 [1p]

Which one of the following statements (A, B, C, D, E) is true?

A. The if-statement is used for iteration, the switch-statement is used for selection
B. The if-statement is used for selection, the switch-statement is used for iteration
C. The if-statement makes choices based on logical conditions, the switch-statement makes choices based on value
D. The if-statement makes choices based on value, the switch-statement makes choices based on logical conditions
E. None of the above is correct

The if-statement refers to the if – else if – else construction. The switch-statement refers to the switch – case construction.

Question 2 [2p]

Below you will find three different programs containing loops. Two of the programs generate the same result, witch of the programs (A, B or C) will not generate the same results as the other two?

A

```c
#include <stdio.h>

int main(void)
{
    int i = 1, sum = 0;
    while(sum <= 20)
    {
        sum = sum + i;
        if(sum%2 == 0)
            i = sum + 1;
        else
            i++;
        printf("%d   ", sum);
    }
    return 0;
}
```

B

```c
#include <stdio.h>

int main(void)
{
    int i = 1, sum = 0;
    do{
        sum = sum + i;
        if(sum%2 == 0)
            i = sum + 1;
        else
            i++;
        printf("%d   ", sum);
    }while(sum <= 20);
    return 0;
}
```

C

```c
#include <stdio.h>

int main(void)
{
    int i, sum;
    for(i = 1, sum = 0; sum <= 20; sum = sum + i)
    {
        if(sum%2 == 0)
            i = sum + 1;
        else
            i++;
         printf("%d   ", sum);
    }
    return 0;
}
```

Correct answer: C

Question 3 [4p]

For each of the assignments below, please indicate how the declaration of the requested variable must look like for the program to compile and work as intended.

Assume the following defined type.

```
struct date{
    int year,month,day;
};
```

Assume that the following declarations exist in the program.

```
struct date toDay, *anyDay;
int y, *u;
float arr[10], z;
```

Example:

Assignment:	Write declaration for
`y = func(6.3);`	`func()` Answer: `int func(float x);`

Explanation: a floating-point number is assigned to d, d must be a float-type variable (the name you use (x) is not important.

Task:

Assignment:	Write declaration for:
`1) print(anyDay, "some date", arr);`	`print()`

`void print(struct date *d, char *str, float arr[]);`
`Ok med char str[], char str[10] eller void *str`
`Ok med float *arr`	

`2) z = foo(&u, arr[0]);`	`foo()`
`float foo(int **x, float y);`	
`3) toDay.month = compute(&y, *anyDay, 8);`	`compute()`
`int compute(int *y, struct date d, int x);`	
`4) toDay = start();`	`start()`
`struct date start(void);`	

Question

Assume the following program:

```
#include <stdio.h>
#include <stdlib.h>

typedef struct
{
    char weekday[10];
    int year, month, day;
}date;

void addData(date *days, int numberOfDays);
void writeToFile(char *fileName, date toDay[], int numberOfDays);

int main(void)
{
    date days[7];
    addData(days, 7);
    writeToFile("file.txt", days, 7);
    return 0;
}

void addData(date *days, int numberOfDays)
{
    //UPPGIFT A
}
void writeToFile(char *fileName, date toDay[], int numberOfDays)
{
    FILE *fp;
    int i;
    fp = fopen(fileName, "w");
    if(fp != NULL)
    {
        fprintf(fp, "%d\n", numberOfDays);
        for(i = 0; i < numberOfDays; i++)
            fprintf(fp, "%s\n%d %d %d\n", (toDay[i]).weekday, (toDay[i]).year,
                                          (toDay[i]).month, (toDay[i]).day);
    }
    fclose(fp);
}
```

Task 4.1 (2p)

Implement the function `addData`. You must use the function declaration in the program above and you are not allowed to do any changes to the existing program.
`addData` should fill the array `days` with information about weekdays, the entire array needs to be filled. The input is made by the user and you need to give instructions so that the user knows what to do.

Solution

```
void addData(date *days, int numberOfDays)
{
    int i;
    for(i = 0; i < numberOfDays; i++) (1p)
    {
        printf("day: ");
        gets((days+i)->weekday);
                                        (1p)
        printf("year, month, day: ");
        scanf("%d %d %d", &(days+i)->year, &(days+i)->month, &(days+i)->day);
        while(getchar()!= '\n');
    }
}
```

Task 4.2 (3p)

The function `writeToFile` saves the content in the array `days` in the text file `file.txt`. Create a new function that saves **the same** information as `writeToFile` but to a binary file instead of a text file. The functions name must be `writeToBinaryFile`. You need to create the function declaration, implementation and show how a call (from main) to the function will look like.

Below you can see the function declaration for the function fwrite( )

```
size_t fwrite(const void *ptr, size_t size, size_t count, FILE *stream);
```

`size_t` is an unsigned int

Solution

```
void writeToBinaryFile(char *fileName, date toDay[], int numberOfDays) //date *toDay (0.5p)
{
    FILE *fp;
    fp = fopen(fileName, "w"); (1p)
    if(fp != NULL)
    {
        fwrite(&numberOfDays, sizeof(int), 1, fp);  (1p)
        fwrite(toDay, sizeof(date), numberOfDays, fp);
    }
}

Anrop: writeToBinaryFile("fil", days, 7); (0.5p)
```

Task 4.3 (2p)

`days` in the program above is an array of type `date`. Assume that the program declares (in main) a pointer to date instead of an array of dates (se below). Create a new function that dynamically allocates memory for a number of dates. The number of dates you need memory for should be given as a parameter to the function. You need to create the function declaration, implementation and show how a call (from main) to the function will look like. You should be able to work with the dynamically allocated memory outside of the function.

```
~~date days[7];~~

date *days = NULL;
```

Solution

```
date *allocateMemory(int numberOfDays)
{
    date *days = (date *)malloc(sizeof(date)*numberOfDays);
    //date days = (date *)calloc(sizeof(date), numberOfDays);
    return days;
}
```

Or

```
void allocateMemory(date **days, int numberOfDays) (0.5p)
{
    *days = (date *)malloc(sizeof(date)*numberOfDays); (1p)
    //*days = (date *)calloc(sizeof(date), numberOfDays);
}

Anrop:  days = allocateMemory(7); (0.5p)

        Or

        allocateMemory(&days, 7);
```

Question 5 [1p]

Use one sentence to describe what a pointer is.

Answer

A pointer is a variable (pointer variable) that contains an address (pointer value) to somewhere in the memory.

Question 6 [4p]

Assume the following declarations.

```
#include <string.h>
#include <stdlib.h>
#include <time.h>

int func(char *str, int x, int *y);

int random, x = 4, y = 9, result;
float arr[10];
char str1[20], *str2, z = '\0';
srand(time(0));
```

Write down the following lines of code (A-L) on your paper and then mark which ones are not allowed/correct. You also need to motivate why they are not allowed/correct.
Points will only be given if a correct motivation is given.

```
A. random = rand()%100 //Missing ;  (0.5p)
B. x + y = result; //Expression must be on right hand side of '=' (0.5p)
C. y+=result;
D. str1 = "programmering"; //Array must be assigned element by element (0.5p)
E. strcpy(str2, "med C");
F. str1[10] = z;
G. arr[x] = arr[++y]; //++y increments y before it's used, try to access index
   10 in array of size 10 – index out of bounds (0.5p)
H. int *p = &x;
I. result = func(&str2, y, *x); //first parameter is char*, str2 is a char* –
   no need for &, last parameter is an integer pointer, x is an integer – send
   the address of x (&x) (1p)
J. result = strlen(str1);
K. puts(str2); //stdio.h is not included (0.5p)
L. str1 = str2; //array type (str1) is not assignable  (0.5p)
```

Uppgift 7 [3p]

Describe (with one scentence each) the three steps that are taken when building your project.

- The preprocessor
- The compiler/compilation
- The linker

Answer

The preprocessor is the first step where all #-lines are taken care of (libraries are included and macros are expanded) (1p)
The compiler checks that the code is correct with regards to syntax and performs optimations on the code, an o-file is created for each c-file compiled. (1p)
The linker generates one .exe-file out of all the o-files and library-files. (1p)