Tentamen - Programmering

DVA117

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

Akademin för innovation, design och teknik

Torsdag 2019-11-07

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

Skrivtid: 08.10 – 13.30
Hjälpmedel: Valfritt icke-elektroniskt material
Lärare: Caroline Uppsäll
(kan nås på telefon om du frågar tentavakten)

Preliminära betygsgränser

Betyg 3:	20p
Betyg 4:	27p
Betyg 5:	33p
Max:	**37p**

Allmänt

- All kod skall skrivas i C.
- Skriv tydligt vilken uppgift/deluppgift ditt svar anser.
- Skriv endast på bladets **ena** sida.
- Referera inte mellan olika uppgifter.
- 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.
- Potentiellt intjänade bonuspoäng kan endast användas på kursinstansens ordinarie tentamen, alltså på denna tentamen för studenter som läser kursen HT19 Period 1.
- Förklaring som till viss del är rätt och till viss del är fel ger inga poäng. Otillräcklig förklaring/beskrivning ger inga poäng.

Lycka till!
/Caroline

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

Notera: alla lösningar bedöms individuellt och förändringar från nedanstående poängsättning (inom uppgiften) kan förekomma beroende på hur resterande svar ser ut.

Uppgift 1 [8p]

Nedan följer 8 kortare frågor. Svara på varje fråga med maximalt 3 meningar vardera.

a) Hur gör man för att allokera minne på heapen?
b) Den pekare som pekar på det dynamiskt allokerade minnet, finns den på stacken eller på heapen?
c) Hur länge existerar minne allokerat på heapen respektive på stacken?
d) Vilket steg i kompileringen är det som expanderar makron?
e) Hur kan man göra för att returnera flera saker ur en funktion?
f) Vad innebär tecknet '\0' och varför är tecknet viktigt?
g) Vad används operatorn == till?
h) Ge ett förslag på ett fall där man starkt bör överväga att använda en do-while konstruktion som loop.

Svar *[1p per rätt svar]*:

a) Man använder dynamisk minnesallokering och allokerar minne med hjälp av funktionen malloc eller calloc.
b) Stacken
c) På heapen existerar minnet tills programmeraren manuellt lämnar tillbaka det (eller tills programmet avslutas). På stacken existerar minnet inom sitt block. – OBS: två svar
d) Preprocessorn *(har man endast svarat att det är i första steget så ger det 0.5p)*
e) Man använder pekare i parameterlistan
f) Strängslut, det är viktig för alla funktioner som itererar igenom strängen- för att veta var strängen tar slut. T.ex. utskrift av strängen eller strlen. - OBS: två svar
g) Jämförelseoperatorn – jämför om två värden är lika
h) Om loopkroppen ska köras minst en gång – tex. Om det finns en meny och användaren ska ange ett menyval. *(har man endast beskrivit den första delen av svaret – utan förslag så ger det 0.5p)*

Uppgift 2 [3p]

Nedanstående program innehållet ett antal fel. Din uppgift är att hitta och rätta felen. Programmet ska slumpa 10 tal i intervallet 50 till 100 och spara dem i arrayen `numbers`. När ett tal har slumpats ska det testas om talet är jämnt eller udda. Information om huruvida talet är jämnt eller udda ska skrivas ut på skärmen och alla jämna och udda tal ska summeras var för sig och de båda summorna ska skrivas ut i slutet av programmet.

Skriv radnumret där du hittar ett fel följt av hur satsen ska se ut.

```
1.    #include <stdio.h>
2.    #include <stdlib.h>
3.    #include <time.h>

4.    int main(void)
5.    {
6.       srand(time(0));
7.       int numbers[10], sumEven, sumOdd;

8.       for(i = 0; i <= 10; i++)
9.       {
10.          numbers[i] = rand()%50+50;
11.          if(numbers[i]%2 == 1)
12.          {
13.              printf("The number %d is even\n", numbers[i]);
14.              sumEven = sumEven + numbers[i];
15.          }
16.          else
17.              printf("The number %d is odd\n", numbers[i]);
18.              sumOdd = sumOdd + numbers[i];
19.       }
20.       printf("The sum of the even numbers are: %d\n", sumEven);
21.       printf("The sum of the odd numbers are: %d\n", sumOdd);
22.       return 0;
23.    }
```

Svar *[0.5p per rätt svar]*:
7 – sumEven och sumOdd måste nollställas
8 – i är inte deklarerad
8 – ska vara i < 10 eller i <= 9
10 – ska vara %51+50
11 – ska vara == 0
16 till 18 – måste vara innanför {}

Uppgift 3 [3p]

Skriv färdigt programmet nedan.

Programmet ska skriva ut ett antal '+' på skärmen. Användaren anger hur många rader plusstecknena ska skrivas ut på samt hur många plus som ska skrivas på varje rad.
Se exempelkörning.

Exempelkörning:

```
How many rows: 3
How many '+' in each row: 6
 +  +  +  +  +  +
 +  +  +  +  +  +
 +  +  +  +  +  +
```

```
#include <stdio.h>

int main(void)
{
    int rows, plusesOnEachRow;

    printf("How many rows: ");
    scanf("%d", &rows);
    printf("How many '+' in each row: ");
    scanf("%d", &plusesOnEachRow);

    /*det är den koden som ska ligga här
          som du ska skriva*/

}
```

Svar:

```
    for(int i = 0; i < rows; i++) [1p]
    {
        for(int k = 0; k < plusesOnEachRow; k++) [1p – måste vara nästlad]
        {
            printf(" + "); [0,5p]
        }
        printf("\n"); [0,5p]
    }
```

Uppgift 4 [11p]

Du ska nu jobba med en del av ett system för ett hotell.

a) [2p] Börja med att definiera en ny typ för ett rum (`room`). Varje rum ska innehålla information om rummets nummer (`roomNr`), antalet sängar i rummet (`nrOfBed`), pris per natt (`pricePerNight`), vilken standard rummet har (`standard` - som en char, 'E' för Economy, 'F' för Family, 'D' för Delux), samt pekare till en gäst (`guest`) som senare kommer vara dynamiskt allokerad. Du ska självklart använda genomtänkta typer på samtliga fält i den nya typen. Som hjälp har du nedanstående typ:

```
struct Person{
    char firstName[30];
    char lastName[30];
    int creditCardNr;
};
```

```
struct room{
    int roomNr;
    int nrOfBeds;
    float pricePerNight;
    char standard;
    struct Person *guest;
};
```

b) [1p] Skapa en dynamiskt allokerad array (`hotel`) som har plats för 100 rum.

```
struct room *hotel = (struct room*)malloc(100*sizeof(struct room));
```

c) [4p] Nedan ser du en funktion som lägger till en ny gäst till ett rum. Funktionen tar som argument adressen till det rum som ska uppdateras. Om rummet inte redan har en gäst (alltså om `guest` är NULL) så allokeras minne för en ny gäst och om allokeringen lyckas får användaren ange information om gästen (förnamn, efternamn och kreditkortsnummer). Din uppgift är att ange vad som ska stå istället för ###1### (samma på två ställen). ###2###, ###3###, ###4### och ###5###

Vi antar att namnen inte innehåller några mellanslag och att en gäst antingen är satt till korrekta värden eller till NULL (om rummet inte har någon gäst).

```
void addNewGuest(struct room *roomToUpdate)
{
    if (roomToUpdate->guest == NULL)
    {
        roomToUpdate->guest = (###1###*)malloc(sizeof(###1###));
        if(###2### != NULL)
        {
            printf("Enter first name: ");
            scanf("%s", ###3###);
            printf("Enter last name: ");
            scanf("%s", ###4###);
            printf("Enter credit card number: ");
            scanf("%d", ###5###);
        }
    }
}
```

Svar:
###1### – struct Person *[1p]*
###2### – roomToUpdate->guest *[1p]*
###3### – roomToUpdate->guest->firstName *[0.5p]*
###4### – roomToUpdate->guest->lastName *[0.5p]*
###5### – &roomToUpdate->guest->creditCardNr *[1p]*

d) [4p] Skapa en funktion som sparar en gästs information till en binär fil. Du ska använda följande funktionshuvud:

```
void saveHotelLogToFile(struct room *roomToLogg, char *fileName);
```

`roomToLogg` är det rum vars gäst ska sparas till loggen. `fileName` är namnet på filen. Din uppgift är att först skriva hotellrummets nummer och sedan all gästens information i binärt format till filen.

Svar:

```
void saveHotelLogToFile(struct room *roomToLogg, char *fileName)
{
    FILE *fp = fopen(fileName, "wb"); [1p]
    if(fp != NULL) [0.5p]
    {
        fwrite(&roomToLogg->roomNr, sizeof(int), 1, fp); [1p]
        fwrite(roomToLogg->guest, sizeof(struct Person), 1, fp); [1p]
    }
    fclose(fp); [0.5p]
}
```

Uppgift 5 [3p]

Hur ser utskriften ut när följande program kört färdigt?

```
#include <stdio.h>
#include <string.h>
#define SIZE 12

void func(char *string1, char *string2)
{
    int k = 0;
    for(int i = strlen(string1)-1; i >= 0; i-=2)
    {
        if(k < strlen(string2))
        {
            string1[i] = *(string2+k);
            k++;
        }
    }
}

int main(void)
{
    char str1[SIZE] = "Programming";
    char str2[SIZE] = "DVA117";

    func(str1, str2);
    puts(str1);
    puts(str2+3);

    return 0;
}
```

Svar:
7r1g1aAmVnD [2p]
117 [1p]
Missat enterslaget –0.5p

Uppgift 6 [5p]

Nedan hittar du ett antal satser. Vissa av dessa är korrekta och andra är antingen felaktiga i syntax eller inte korrekta trots att de accepteras av kompilatorn.

Din uppgift är att hitta de satser som på något sätt är felaktiga samt förklara varför de är felaktiga. Skriv av numret för de satser du anser är felaktiga på ditt svarspapper och förklara varför de är felaktiga.

Har du angivit att någon korrekt sats är felaktig så kommer det svaret att kvittas mot ett korrekt angivet svar.

Du har följande inkuderingar, definitioner och deklarationer. Alla satser (1-20) är fristående från varandra.

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

```c
struct city{
    char name[30];
    int inhabitants;
    float longitude;
};

int sum(int tal1, int tal2);
void funk1(struct city *info);
struct city funk2(char t, float f);

int main(void)
{
    char *str = "ProgRammering";
    struct city myCity;
    struct city *pmyCity = &myCity;
    int arr[10];
    int i = 9, a = 5, b = 3, *pNumber = &a;
    float x = -1.5, y = 10.0;
    char letter, *pLetter = &letter;

    ...

    return 0;
}
```

```c
1.     i = sum(a, b);
2.     myCity.longitude = y;
3.     switch(x){...} //måste vara heltal
4.     a + b = i; //fel håll på tilldelning
5.     strcpy(myCity.city, "Stockholm"); //city är inte ett fält i strukten
6.     myCity = (struct city*)calloc(10, sizeof(struct city));//pekare
7.     x = x * *pNumber;
8.     printf("%s", str);
9.     arr[++i] = 3; //i är 10, utanför arrayen
10.    *pmyCity.inhabitants = 153000; //parenteser saknas
11.    i = sum(*pNumber, b);
12.    str[4] = 'r'; //kan inte ändra element i konstansträng
13.    funk1(&pmyCity); //ta bort &
14.    scanf("%d", arr+2);
15.    myCity = funk2(letter); //för få argument
16.    pmyCity->inhabitants = 153000;
17.    *letterp = 'a';  //letterp finns inte
18.    arr = pNumber; //får inte flytta arraypekare
19.    *(arr+1) = *(arr+2);
20.    i = sum(a, sum(a, b));
```

Svar:
3. 4. 5. 6. 9. 10. 12. 13. 15. 17. 18
OBS! felet på uppgift 5 var ett misstag men ändock ett fel så totalpoängen på uppgiften är justerad till 5.5p. Vilket göra att totalpoängen på tentamen är 37.5p

Uppgift 7 [4p]

Antag följande struktdefinitioner:

```
struct fish{
    char name[30];
    int nrOfTeeth;
    int age;
};

struct aquarium{
    struct fish[100];
};
```

Hur mycket mycket minne tas i anspråk när:

a) variabeln `struct fish snappy;` deklareras? //30+4+4 = 38byte
b) strukten `struct fish` definieras? //0byte
c) variabeln `struct aquarium myAquarium;` deklareras? //100*38 = 3800byte
d) variablen `struct fish *myFavorite = &snappy;` deklareras? //8 byte (eller 4 byte beroende på system)- en pekare

FACIT /SOLUTIONS

Exam - Programming

DVA117

----------Solutions----------

School of Innovation, design and technology

Thursday 2019-11-07

Writing time: 08.10 – 13.30
Aids: Any non-electronic material
Examiner: Caroline Uppsäll
(Can be reached by telephone if you ask the exam guard)

Preliminary grading limits
Grade 3: 20p
Grade 4: 27p
Grade 5: 33p
Max: 37p

Generally

- All code should be written in C.
- Write clearly what task/sup-task your answers consider.
- Do only use **one** side of the paper.
- Do not refer between questions.
- 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.
- Explanations that are to some extent correct and to som extent wrong does not give any points. Insufficient explanations/descriptions gives no points.

----------Solutions----------

Good luck!
/Caroline

---------- Solutions ----------

Note: all solutions are assessed invidually and changes from the following scoring (within the task) my occur depending on what the remaining answer looks like.

Question 1 [8p]

Below you will find 8 short questions. Answer each question with a maximum of 3 sentences.

a) How would you do if you want to allocate memory on the heap?
b) Is the pointer that points to the dynamically allocated memory, allocated on the stack or on the heap?
c) For how long will memory allocated on the heap exist and for how long will memory allocated on the stack exist?
d) Which step in the compilation will take care of the macro expansion?
e) How can you solve the problem of having to return multiple values from a function?
f) What does the character '\0' mean and why is it important?
g) What is the operator == used for?
h) Give and example of a case where one strongly should consider using a do-while construction for the loop logic.

Answer *[1p for each correct answer]*:

a) You use dynamic memeoryallocation by using the function malloc or calloc.
b) The stack
c) On the heap the memory exists until it is manually freed (or until the program closes). On the stack the memory only exists in the scope/block where it is created – NOTE: two answers.
d) The pre processor
e) You use pointers in the list of parameters.
f) End of string, it is important for all functions that iterate through the string – to know where the string ends. For example printing the string or strlen(). – NOTE: two answers.
g) Comparison – compares if the two sides are equal.
h) If the statements in the loop whould run at least once – for example if there is a menu and the user are supposed to make a choice.

Question 2 [3p]

The program below contains a number of errors. Your task is to find and correct the errors. The program should randomly genereate 10 numbers in the interval 50 to 100 and save them in the array `numbers`. When a number has been generated it should be tested if it is even or odd. Information about if it is even or odd should be printed on the screen and all even and odd numbers should be summarized separately. The total of the even numbers and the total of the odd numbers should be printed on the screen at the end of the program.

Write down the number of the row where you find the error followed by how to correct the statement.

```
1.    #include <stdio.h>
2.    #include <stdlib.h>
3.    #include <time.h>

4.    int main(void)
5.    {
6.       srand(time(0));
7.       int numbers[10], sumEven, sumOdd;

8.       for(i = 0; i <= 10; i++)
9.       {
10.          numbers[i] = rand()%50+50;
11.          if(numbers[i]%2 == 1)
12.          {
13.              printf("The number %d is even\n", numbers[i]);
14.              sumEven = sumEven + numbers[i];
15.          }
16.          else
17.              printf("The number %d is odd\n", numbers[i]);
18.              sumOdd = sumOdd + numbers[i];
19.       }
20.       printf("The sum of the even numbers are: %d\n", sumEven);
21.       printf("The sum of the odd numbers are: %d\n", sumOdd);
22.       return 0;
23.    }
```

Answer *[0.5p for each correct answer]*:
7 – sumEven and sumOdd has to be initialized to 0
8 – i is not declared
8 – should be i < 10 or i <= 9
10 – should be %51+50
11 – should be == 0
16 to 18 – add { }

Question 3 [3p]

Finish the program below.

The programs task is to print a number of '+' on the screen. The user will input how many rows there will be and also how many '+' there should be on each row.
See example.

Example:

```
How many rows: 3
How many '+' in each row: 6
 +  +  +  +  +  +
 +  +  +  +  +  +
 +  +  +  +  +  +
```

```
#include <stdio.h>

int main(void)
{
    int rows, plusesOnEachRow;

    printf("How many rows: ");
    scanf("%d", &rows);
    printf("How many '+' in each row: ");
    scanf("%d", &plusesOnEachRow);

        /*Your code goes here*/

}
```

Answer:

```
    for(int i = 0; i < rows; i++) [1p]
    {
        for(int k = 0; k < plusesOnEachRow; k++) [1p – nested]
        {
            printf(" + "); [0,5p]
        }
        printf("\n"); [0,5p]
    }
```

Question 4 [11p]

You will now work on a system for a hotel.

a) [2p] Start by defining a new type that will hold information about a room (`room`). Each room should hold information about the rooms number (`roomNr`), the number of beds in the room (`nrOfBed`), price for each night (`pricePerNight`), the standard of the room (`standard` – as a char, 'E' for Economy, 'F' for Family, 'D' for Delux), and a pointer to the guest staying in the room (`guest`). The guest will later be dynamically allocated. You should of course use well thought out types in all fields of the new type. To help, you have the following type:

```
struct Person{
    char firstName[30];
    char lastName[30];
    int creditCardNr;
};
```

```
struct room{
    int roomNr;
    int nrOfBeds;
    float pricePerNight;
    char standard;
    struct Person *guest;
};
```

b) [1p] Create a dynamically allocated array (`hotel`) that has 100 rooms.

```
struct room *hotel = (struct room*)malloc(100*sizeof(struct room));
```

c) [4p] Below you will find a function that adds a new guest to a room. The function take, as an argument, the address for the room to be updated. If the room does not already have a guest (if `guest` is NULL) memory will be allocated for a new guest. If the allocation was successful the use will be asked to give information about the guest (first name, last name and credit card number). Your task is to say what code should be written instead of ###1### (the same in two spots). ###2###, ###3###, ###4### and ###5###

We assume that the name never contains spaces and that a guest is ether set to correct values or NULL (if there is no guest in the room).

```
void addNewGuest(struct room *roomToUpdate)
{
    if (roomToUpdate->guest == NULL)
    {
        roomToUpdate->guest = (###1###*)malloc(sizeof(###1###));
        if(###2### != NULL)
        {
            printf("Enter first name: ");
            scanf("%s", ###3###);
            printf("Enter last name: ");
            scanf("%s", ###4###);
            printf("Enter credit card number: ");
            scanf("%d", ###5###);
        }
    }
}
```

Answer:

```
###1### - struct Person [1p]
###2### - roomToUpdate->guest [1p]
###3### - roomToUpdate->guest->firstName [0.5p]
###4### - roomToUpdate->guest->lastName [0.5p]
###5### - &roomToUpdate->guest->creditCardNr [1p]
```

d) [4p] Create a function that saves a guests information to a binary file. You must use the function declaration below:

```
void saveHotelLogToFile(struct room *roomToLogg, char *fileName);
```

`roomToLogg` is the room where the guest you are suppose to save to the file is staying. `fileName` is the name of the file.

Your task is to first save the room number to the file and then all the guest information to the file (in binary form).

Answer:

```
void saveHotelLogToFile(struct room *roomToLogg, char *fileName)
{
    FILE *fp = fopen(fileName, "wb"); [1p]
    if(fp != NULL) [0.5p]
    {
        fwrite(&roomToLogg->roomNr, sizeof(int), 1, fp); [1p]
        fwrite(roomToLogg->guest, sizeof(struct Person), 1, fp); [1p]
    }
    fclose(fp); [0.5p]
}
```

Question 5 [3p]

What will the output look like when the following program is done executing?

```
#include <stdio.h>
#include <string.h>
#define SIZE 12

void func(char *string1, char *string2)
{
    int k = 0;
    for(int i = strlen(string1)-1; i >= 0; i-=2)
    {
        if(k < strlen(string2))
        {
            string1[i] = *(string2+k);
            k++;
        }
    }
}

int main(void)
{
    char str1[SIZE] = "Programming";
    char str2[SIZE] = "DVA117";

    func(str1, str2);
    puts(str1);
    puts(str2+3);

    return 0;
}
```

Answer:
7r1g1aAmVnD
117

Question 6 [5p]

Below you will find a number of statements. Some of them are correct and some are ether wrong in syntax or incorrect even though they are accepted by the compiler.

Your task is to find the statements that in some sence are incorrect and explain why they are incorrect. Write down the number of the statemens you state are incorrect and give an explanation.

If you state a correct statement as incorrect that answer will be offset against a correct answer.

You have the following includes, definitions och declarations. All statements (1-20) are independent of each other.

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

```c
#include <stdlib.h>


struct city{
    char name[30];
    int inhabitants;
    float longitude;
};

int sum(int tal1, int tal2);
void funk1(struct city *info);
struct city funk2(char t, float f);

int main(void)
{
    char *str = "ProgRammering";
    struct city myCity;
    struct city *pmyCity = &myCity;
    int arr[10];
    int i = 9, a = 5, b = 3, *pNumber = &a;
    float x = -1.5, y = 10.0;
    char letter, *pLetter = &letter;

    ...

    return 0;
}
```

```c
1.    i = sum(a, b);
2.    myCity.longitude = y;
3.    switch(x){...} //must be integer
4.    a + b = i; //wrong direction of =
5.    strcpy(myCity.city, "Stockholm"); //city is not a member in struct
6.    myCity = (struct city*)calloc(10, sizeof(struct city));//pointer
7.    x = x * *pNumber;
8.    printf("%s", str);
9.    arr[++i] = 3; //i is 10, out of bounds
10.   *pmyCity.inhabitants = 153000; //brackets missing
11.   i = sum(*pNumber, b);
12.   str[4] = 'r'; //can not change elements in a constant string
13.   funk1(&pmyCity); //remove &
14.   scanf("%d", arr+2);
15.   myCity = funk2(letter); //to few arguments
16.   pmyCity->inhabitants = 153000;
17.   *letterp = 'a';  //letterp is not declared
18.   arr = pNumber; //not allowed to move array-pointer
19.   *(arr+1) = *(arr+2);
20.   i = sum(a, sum(a, b));
```

Answer:
3. 4. 5. 6. 9. 10. 12. 13. 15. 17. 18

Question 7 [4p]

Assume the following struct definitions:

```
struct fish{
    char name[30];
    int nrOfTeeth;
    int age;
};

struct aquarium{
    struct fish[100];
};
```

How much memory will be allocated when…

a) the variable `struct fish snappy;` is declared? //30+4+4 = 38byte
b) the struct `struct fish` is defined? //0byte
c) the variable `struct aquarium myAquarium;` is declared? //100*38 = 3800byte
d) the variable `struct fish *myFavorite = &snappy;` is declared? //8 || 4 byte a pointer