King Abdulaziz University
Department of Computer Science

CS 201 - Introduction to Computer Science

Spring 2007

Draft 0 Revision 4
February 20, 2007

توزيع درجات النشاط العملي

الامتحان العملي	١٥
واجب مذكرة المعمل والواجب النظري (أنظر مفردات المقرر)	٥

يهدف الامتحان العملي للتأكد من أنك قضيت الساعات المطلوبة في النشاط العملي، وأديت النشاط بالعناية المتوقعة. الامتحان مدته ٣٥ دقيقة في الاسبوع الاخير للدراسة في وقت المعمل. سوف يطلب منك كتابة وتشغيل برنامج بسيط. يسمح باستعمال الكتاب ومذكرة المعمل أثناء الامتحان.

You may find some of the Flash demos in www.hashimi.ws/201 useful.

أخي الطالب

نضع بين يديك مذكرة معمل حاسب٢٠١ مقدمة لعلم الحاسوب. يهدف النشاط في هذه المذكرة لاكسابك المهارات العملية لبرمجة الحاسوب. وهي مهارات مفيدة في دراستك العلمية وحياتك العملية، بصرف النظر عن تخصصك.

المذكرة مصممة باسلوب التعلم الاكتشافي عن طريق التجربة والمثال. أي يتوقع منك الملاحظة والنسج على منوال الامثلة المدرجة في النشاط. والمعرفة التي تهدف المذكرة لتنميتها معرفة مهارية تراكمية تستغرق ما لا يقل عن ١٦-٢٠ ساعة خلال الفصل. بمعنى أنك لن تستطيع ببساطة "المذاكرة" للامتحان العملي ليلة الامتحان.

لذلك، ولتحصل على أكبر فائدة من النشاط العملي ننصحك بما يلي:

- عمل كل نشاط كاملا وبعناية.
- اتباع خطوات النشاط وعمل المطلوب في كل خطوة بانتباه مع الحرص على اجابة الأسئلة المدرجة وتدوين الملاحظات العملية في المكان المخصص.
- إذا واجهت صعوبة في أداء النشاط فلا تتردد في سؤال المشرف على المعمل وطلب العون منه مبكرا لتخطي الصعوبات أولا بأول.

أخيرا وليس آخرا، لا تدع المذكرة تحد من مخيلتك وابداعك. جرب واستعن بالكتاب المقرر (خصوصا الفصل التاسع)، وبخدمات المساعدة في برنامج Visual Studio.NET، واسأل إذا احتجت.

يتوقع أن يستغرق كل نشاط معملي ما لا يقل عن ساعتين أسبوعيا. مع معظم الحصص يوجد واجب منزلي بسيط يتوقع تسليمه الحصة التالية لمشرف معملك.

أخي الطالب، اهتمامك بأداء النشاط العملي جزء كبير من استفادتك من حاسب٢٠١، وسوف يؤدي بإذن الله إلى اكتسابك مهارة تستمر معك مدى الحياة.

مع تحيات

أسرة معمل حاسب ٢٠١

Lab 1

Objectives

Key terms are in **bold**. Review definitions of these terms in Textbook Section 9.5.

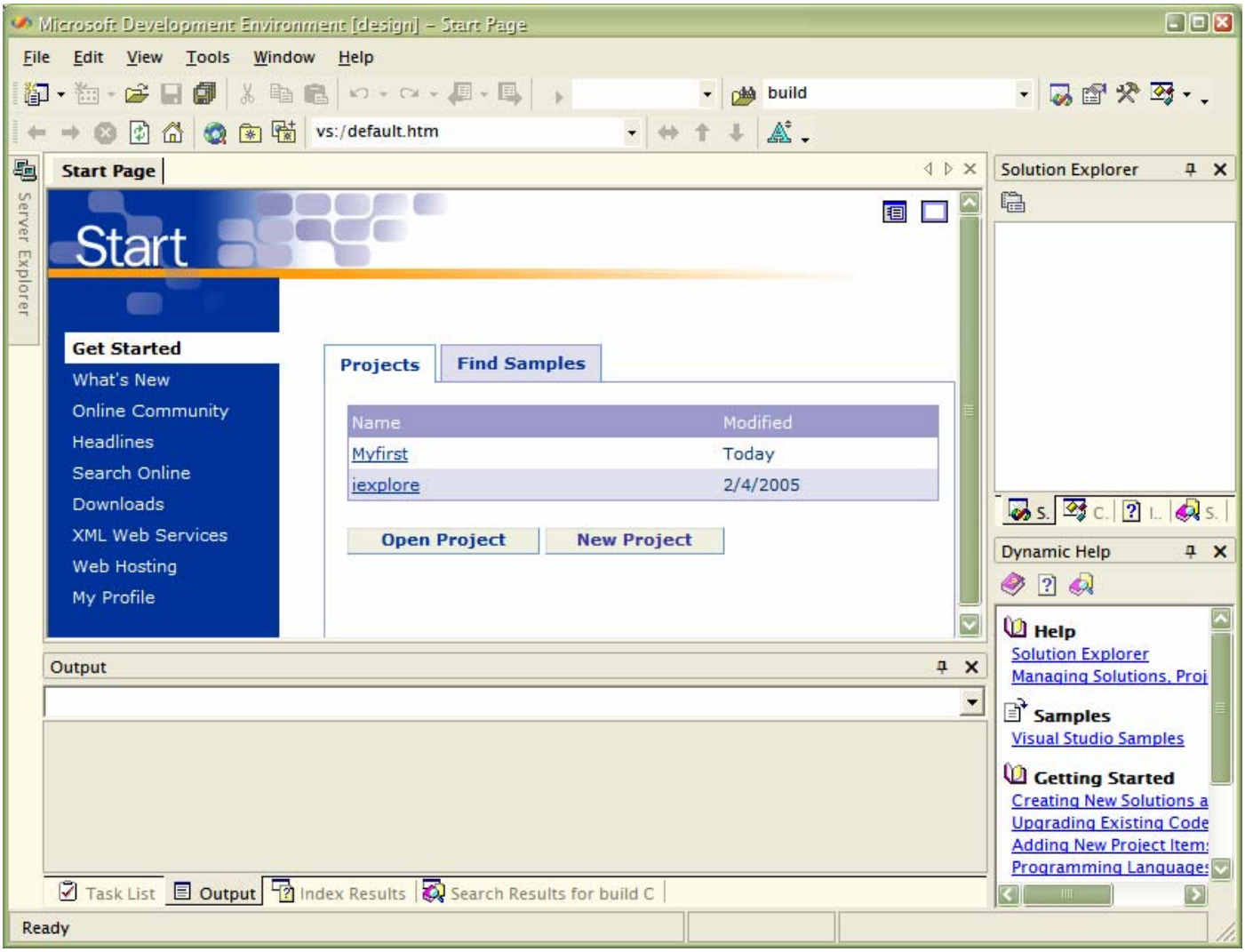
- ☑ Describe the **Windows Desktop** metaphor (what does the term Desktop mean?).
- ☑ Familiarize with: Windows Desktop, Start menu (All Programs), and My Documents.
- ☑ Files: data files (for example: .C or .TXT) and program files (.EXE).
- ☑ Windows: open programs (run .EXE), open data files (in view in Notepad).
- ☑ Locate (find) C programs.
- ☑ Create/delete/rename folder.
- ☑ Locate (find) and run IDE.
- ☑ Open program in IDE.
- ☑ Save program.
- ☑ Change program name using Save As.
- ☑ Get help from your IDE: Help Menu, and context help using F1.

Student Notes

Lab Sheet

1. Your Visual Studio.NET screen should look similar to the screenshot below.

Notice the project Myfirst in the **Projects** area. You will open this project and work on the program stored in it.



Lab 2

Objectives

Key terms are in **bold**. Review definitions of these terms in Textbook Section 9.5.

- ☑ Load program in IDE.
- ☑ Basic **edit**, **compile** and **run** (execute) cycle.
- ☑ Practice with: Save program, change name using Save As, and using the IDE help features.
- ☑ Organize your programs files in folders.
- ☑ Move files between your floppy or flash drive and the lab computer.

Student Notes

Lab 3

Objectives

Key terms are in **bold**. Review definitions of these terms in Textbook Section 9.5.

- ☑ Create new program in IDE.
- ☑ Basic **edit**, **compile** and **run** (execute) cycle.
- ☑ **Source** (user) code and **object** (machine) code.
- ☑ **Syntax** errors.
- ☑ String **literal constant**.
- ☑ Number **literal constant**.

Student Notes

Lab Sheet

1. Type the following program carefully including spaces.

```
12345678901234567890123456789012345678901234567890123456789012345678901234567890123456789012
1  #include <stdio.h>
2  main()
3  {
4     printf("Hello World!");
5  }
```

Visit www.hashimi.ws/cs101 and check the lab3 Flash demo for step-by-step demonstration of creating a C program file in your Integrated Development Environment (IDE).

2. Save this program as: hello.c

3. Compile and run.

Write the keyboard commands to do the following:
Save_____ Run_____ Compile Only_____ Execute Compiled Program_____

4. Change line 4 as follows. Compile and run.

```
4     printf("Hi There!");
```

5. Delete line 5 so that your program looks as follows. Compile and run.

```
12345678901234567890123456789012345678901234567890123456789012345678901234567890123456789012
1  #include <stdio.h>
2  main()
3  {
4     printf("Hi There!");
5
```

Write the error message? What does it mean (in Arabic)?

Error message:_____

Meaning:_____

6. Correct the program. Compile and run to check.

7. Change line 4 as follows. Compile and run.

```
4     printf("Hi There!)
```

Write the error message? What does it mean (in Arabic)?

Error message:_____

Meaning:_____

8. Correct the program. Compile and run to check.

9. Change line 4 as follows. Compile and run.

```
4     printf("%d", 12+8);
```

10. Change line 4 as follows. Compile and run.

```
4     printf("12+8”);
```

What is the difference between 9, 10.

Extra Work

11. Type the following program carefully including spaces. Compile and run.

Save As: hello2.c

```
    123456789012345678901234567890123456789012345678901234567890123456789012
1   #include <stdio.h>
2   main()
3   {
4      printf("Hello World!\n");
5      printf("Fine thank you.");
6   }
```

12. Change line 4 as follows. Compile and run.

```
4     printf("Hello World!");
```

What does `\n` do?_____

13. Change line 4 as follows. Compile and run.

```
4     printf("Hello World!\n\n\n");
```

14. Change line 4 as follows. Compile and run.

```
4     printf("%s\n\n\n", "Hello World!");
```

15. Write a program (call it lab1.c) to calculate: 20-32+7. Print the message “The result=” before the answer. Compile and run to test your answer.

(Hint: use information from 9 and 14.)

Name	Student ID

Homework

1. The line `#include <stdio.h>` is an example of **preprocessor directive**. Define the terms: **preprocessor**, and **preprocessor directive**. *(Section 9.2 on pages 169-170 in your textbook).*
2. Study Figure 9.2 (page 170). Draw a similar diagram for what you did in this lab. Write down menu or keyboard command for each step in the diagram. Explain in your own words (Arabic, if you like) the meaning of *Source* and *Object* on the figure.
3. Practice using the Flash demo online at: www.hashimi.ws/cs101. Check the lab3 link under Learning Resources.

Lab 4

Key terms are in **bold**. Review definitions of these terms in Textbook Section 9.5.

- ☑ Integer **variable**s (**identifier**s): `int`
- ☑ Floating point **variable**s: `float`, `double`.
- ☑ **Expressions** and **operator**s: +,-,*,/, ( ).
- ☑ **Assignment** operator.
- ☑ **Statements:** assignment statement.

Student Notes

Lab Sheet

1. Type the following program carefully and save as lab2.c. Compile and run.

Save As: lab2.c

```
    12345678901234567890123456789012345678901234567890123456789012345678901234567890123456789012
1   #include <stdio.h>
2   main()
3   {
4      int x=10;
5      printf("The number is %d", x);
6   }
```

2. Change lab2.c as follows. Compile and run.

```
    12345678901234567890123456789012345678901234567890123456789012345678901234567890123456789012
1   #include <stdio.h>
2   main()
3   {
4      int x=10, y;
5      y=10;
6      printf("The sum is %d", x+y);
7   }
```

3. Change lab2.c as follows. Compile and run.

```
    12345678901234567890123456789012345678901234567890123456789012345678901234567890123456789012
1   #include <stdio.h>
2   main()
3   {
4      int x=10, y;
5      scanf("%d", &y);
6      printf("The difference is %d\n", x-y);
7   }
```

Notice that `scanf` requires '`&`' before the variable name.

4. Type the following program carefully. Compile and run. Test the inputs: 55, 102.

Save As: f2c.c

```
    12345678901234567890123456789012345678901234567890123456789012345678901234567890123456789012
1   #include <stdio.h>
2   main()
3   {
4      int cels, fahr;
5      scanf("%d", &fahr);
6      cels=(fahr-32)*5/9;
7      printf("%d fahrenheit = %d celsius.\n", fahr, cels);
8   }
```

In line 6 we calculate the **expression** $\frac{5}{9}$ (*fahrenheit* - 32).

5. Change f2c.c as follows and save as f2c2.c. Compile and run. Test with the inputs: 55, 102, 102.4.

Save As: f2c2.c

```
12345678901234567890123456789012345678901234567890123456789012345678901234567890123456789012
1  #include <stdio.h>
2  main()
3  {
4     float cels, fahr;
5     scanf("%f", &fahr);
6     cels=(fahr-32)*5.0/9.0;
7     printf("%f fahrenheit = %f celsius.\n", fahr, cels);
8  }
```

What is the difference between results from f2c.c and f2c2.c. Write the output when input is 102.4: _____

6. Change f2c2.c as follows. Compile and run. Test with 102.4.

```
12345678901234567890123456789012345678901234567890123456789012345678901234567890123456789012
1  #include <stdio.h>
2  main()
3  {
4     double cels, fahr;
5     scanf("%lf", &fahr);
6     cels=(fahr-32)*5.0/9.0;
7     printf("%lf fahrenheit = %lf celsius.\n", fahr, cels);
8  }
```

Compare the output of input 102.4 with the same case from the last step.

7. Change lab2.c as follows. Compile and run.

```
12345678901234567890123456789012345678901234567890123456789012345678901234567890123456789012
1  #include <stdio.h>
2  main()
3  {
4     int x=10, y=20;
5     printf("%d\n", x/y);
6  }
```

Try changing `int` to `float` in line 4, and `%d` to `%f` in line 5. What is the new output? Why?

_____

_____

Extra Work

8. Type the following program carefully. Compile and run.

Save As: avetwo.c

```
    123456789012345678901234567890123456789012345678901234567890123456789012
 1  #include <stdio.h>
 2  main()
 3  {
 4     int x1,x2;
 5     printf("Input two numbers: ");
 6     scanf("%d %d", &x1, &x2);
 7     printf("The average of %d and %d is: %f", x1, x2, (x1+x2)/2.0);
 8  }
```

What happens if we remove "`( )`" around `x1+x2` in line 7? Try it.

_____

_____

9. Type the following program carefully. Compile and run.

Save As: swap.c

```
    123456789012345678901234567890123456789012345678901234567890123456789012
 1  #include <stdio.h>
 2  main()
 3  {
 4     int x=7, y=-3, temp;
 5     temp = x;
 6     x = y;
 7     y = temp;
 8     printf("x=%d, y=%d\n", x,y);
 9  }
```

The program uses assignments to swap (exchange) the contents of the variables `x` and `y`.

10. Type the following program carefully. Compile and run.

Save As: hello2.c

```
    123456789012345678901234567890123456789012345678901234567890123456789012
 1  #include <stdio.h>
 2  main()
 3  {
 4     char name[100];
 5     printf("Input your name: ");
 6     scanf("%s", &name);
 7     printf("Hello %s!\n", name);
 8  }
```

The variable type `char` can be used to store characters (string data). In the code above, the variable `name` can store up to 100 characters.

Note: `int`, `float`, `double`, and `char` are standard **types** of variables (definition in Textbook on page 179).

Name	Student ID

Homework

1. When does the **assignment expression** `x=10` become an **assignment statement**? *(Answer: in textbook page 183)*.
2. In the Textbook (pages 180-181), review the definition of the following terms: variable **declaration**, variable **definition**, variable **initialization**. Give examples for each from the programs of this lab.

Lab 5

Key terms are in **bold**. Review definitions of these terms in Textbook Section 9.5.

- ☑ **Reserved** words (**keywords**).
- ☑ **Iterative statement**: `while` **loop**.
- ☑ **Selection** statement: `if` and `else`.
- ☑ Increment **operator**: `++`.
- ☑ Relational **operator**: `<`.

Student Notes

Lab Sheet

1. Type the following program carefully. Compile and run.

Save As: sum.c

```
    123456789012345678901234567890123456789012345678901234567890123456789012
1   #include <stdio.h>
2   main()
3   {
4      int x, sum=0, i=0;
5      while (i<5)
6      {
7         scanf("%d", &x);
8         sum = sum + x;
9         i=i+1;
10     }
11     printf("%d\n", sum);
12  }
```

2. Change sum.c as follows. Compile and run.

```
5      whil (i<5)
```

Write the error message? What does it mean (in Arabic)?

Error message:_____

Meaning:_____

3. Change sum.c to calculate the sum and average. Compile and run.

```
    123456789012345678901234567890123456789012345678901234567890123456789012
1   #include <stdio.h>
2   main()
3   {
4      int x, sum=0, i=0;
5      while (i<5)
6      {
7         scanf("%d", &x);
8         sum = sum + x;
9         i=i+1;
10     }
11     printf("sum=%d\naverage=%f\n", sum, sum/5);
12  }
```

Check the output. What is wrong? (Hint: the **literal constant** `5` in line 11 is `int`).

_____

4. Change sum.c from 2 as follows. Compile and run.

```
11     printf("sum=%d\naverage=%f\n", sum, sum/5.0);
```

Why did the code from 4 not work?_____

5. Change sum.c to process 4 numbers instead of 5. Compile and run.

6. Change sum.c as follows. Compile and run.

```
    123456789012345678901234567890123456789012345678901234567890123456789012
1   #include <stdio.h>
2   main()
3   {
4      int x, sum=0, i=0, n=5;
5      while (i<n)
6      {
7         scanf("%d", &x);
8         sum = sum + x;
9         i=i+1;
10     }
11     printf("sum=%d\naverage=%f\n", sum, (float) sum/n);
12  }
```

Why is the code in 5 better? Previously, did you remember to change 5 to 4 everywhere?

_____

_____

Note: `(float)` in line 11 forces `sum/n` to be real. Another solution is to declare `sum` as `float` (remove `sum` from line 4, add the line `float sum=0.0;` after line 4). Try it!

7. Type the following program carefully. Compile and run.

Save As: passfail.c

```
    123456789012345678901234567890123456789012345678901234567890123456789012
1   #include <stdio.h>
2   main()
3   {
4      int x;
5      scanf("%d", &x);
6      if (x<60)
7         printf("fail\n");
8      else
9         printf("pass\n");
10  }
```

Extra Work

8. Change passfail.c to input and process 5 grades. Compile and run.

```
12345678901234567890123456789012345678901234567890123456789012345678901234567890123456789012
```

```
1  #include <stdio.h>
2  main()
3  {
4     int x, i=0, n=5;
5     while (i<n)
6     {
7        scanf("%d", &x);
8        if (x<60)
9           printf("fail\n");
10       else
11          printf("pass\n");
12       i=i+1;
13    }
14 }
```

Notice how passfail,c (from 8) is similar to sum.c (from 6). The same loop can be used for any processing on the value of `x`.

9. Change sum.c (from 6) as follows. Compile and run.

```
9        i++;
```

The expression `i++` is equivalent to `i=i+1`. They have the same meaning but `i++` is more common.

10. Change sum.c to output the product (multiply) instead of the sum (add). Save as prod.c. Compile and run.

Homework

1. Draw a flowchart for the algorithm used in sum.c from 6.

Lab 6

Key terms are in **bold**. Review definitions of these terms in Textbook Section 9.5.

- ☑ Comments.
- ☑ More **arithmetic operator**s: `%`.
- ☑ More **assignment operator**s: `+=`, `-=`, `*=`, `/=`, `%=`.
- ☑ Decrement **operator**: `--`.
- ☑ **Named constant:** `const`.

Student Notes

Lab Sheet

1. Type the following program carefully. Compile and run.

Save As: sum2.c

```
    123456789012345678901234567890123456789012345678901234567890123456789012
 1  #include <stdio.h>
 2  main()
 3  {
 4     int x, sum=0, i=5;
 5     while (i>0)
 6     {
 7        scanf("%d", &x);
 8        sum = sum + x;
 9        i--;   /* i-- means i=i-1 */
10     }
11     printf("sum=%d\n", sum);
12  }
```

The text between `/*` and `*/` is a **comment** for the programmer. It is ignored by the compiler. Comments are used to explain the program.

This program uses a decreasing loop. Notice how we do no need `n` (compare to sum.c).

2. Change sum2.c as follows. Compile and run.

```
8        sum += x;
```

3. Type the following program carefully. Compile and run.

Save As: sumeven.c

```
    123456789012345678901234567890123456789012345678901234567890123456789012
    /* Program adds even numbers from 0 to number input by user */
 1  #include <stdio.h>
 2  main()
 3  {
 4     int max, i, sum=0;
 5     scanf("%d", &max);
 6     i=max;
 7     while (i>0)
 8     {
 9        if (i%2 == 0)   /* % means the remainder of i/2 */
10           sum += i;    /* sum+=i means sum=sum+i */
11        i--;
12     }
13     printf("%d is the sum of even < or = %d\n", sum, max);
14  }
```

4. Type the following program carefully. Compile and run.

Save As: factx.c

```
    123456789012345678901234567890123456789012345678901234567890123456789012
    /* Program to calculate the factorial of number input by user */
 1  #include <stdio.h>
 2  main()
```

```
3  {
4     int x, i, fact=1;
5     scanf("%d", &x);
6     i=x;
7     while (i>0)
8     {
9        fact *= i;   /* fact*=i means fact=fact*i */
10       i--;
11    }
12    printf("factorial(%d)=%d\n", x,fact);
13 }
```

Write the output when the input is -1: _____

Is the output correct for negative input?_____

Can you modify the program to print an error message and stop when the input is negative?

5. Type the following program carefully. Compile and run. Test with positive integers as input.

Save As: findLargest.c

```
    123456789012345678901234567890123456789012345678901234567890123456789012
1  #include <stdio.h>
2  main()
3  {
4     int x, largest=0, i=0, n=5;
5     while (i<n)
6     {
7        scanf("%d", &x);
8        if (x > largest)
9           largest = x;
9        i++;
10    }
11    printf("largest=%d\n", largest);
12 }
```

This program uses the same algorithm used in your Textbook Algorithm 8.5 on page 149. It assumes that the input numbers are greater or equal 0.

Extra Work

6. Type the following program carefully. Compile and run. Test with positive integers from 0 to 100.

Save As: findSmallest.c

```
    123456789012345678901234567890123456789012345678901234567890123456789012
1   #include <stdio.h>
2   main()
3   {
4      int x, smallest=100, i=0, n=5;
5      while (i<n)
6      {
7         scanf("%d", &x);
8         if (x < smallest)
9            smallest = x;
10        i++;
11     }
12     printf("smallest=%d\n", smallest);
13  }
```

7. Change findSmallest.c as follows (add line 4a after line 4). Compile and run.

```
4      int x, i=0, n=5;
4a     const int smallest=100;
```

8. Add the following after line 12. Compile and run.

```
12a      smallest = 110;
```

Write the error message? What does it mean (in Arabic)?

Error message:_____

Meaning:_____

9. Write a program (carea.c) which asks the user for a number and outputs the area of a circle with the input as radius. Compile and run to test your answer.

(Hint: the area of a circle is πr^2, where r is the radius).

Homework

1. Draw a flowchart for the algorithm used in findSmallest.c from 6.

Lab 7

Key terms are in **bold**. Review definitions of these terms in Textbook Section 9.5.

- ☑ **Iterative statement**: `for` **loop**.
- ☑ **Arrays**.

Student Notes

Lab Sheet

1. Change sum.c as follows (change lines 4, 5 and delete line 9). Compile and run.

Save As: sum3.c

```
    123456789012345678901234567890123456789012345678901234567890123456789012
1   #include <stdio.h>
2   main()
3   {
4      int x, sum=0, i, n=5;
5      for (i=0; i<n; i++)
6      {
7         scanf("%d", &x);
8         sum += x;
9      }
10     printf("sum=%d\n", sum);
11  }
```

The `for`-loop is equivalent to `while`. It is easier to write the loop as `for` when it is controlled by a counter.

2. Change findLargest.c as follows (change lines 4, 5 and delete line 9). Compile and run.

Save As: findLargest2.c

```
    123456789012345678901234567890123456789012345678901234567890123456789012
1   #include <stdio.h>
2   main()
3   {
4      int x, largest=0, i, n=5;
5      for (i=n; i>0; i--)
6      {
7         scanf("%d", &x);
8         if (x > largest)
9            largest = x;
10    }
11    printf("largest=%d\n", largest);
12  }
```

3. Type the following program carefully. Compile and run.

Save As: arr.c

```
    123456789012345678901234567890123456789012345678901234567890123456789012
1   #include <stdio.h>
2   main()
3   {
4      int y[5]={1,2,3,4,5}; /* define and initialize array of 5 integers */
5      int i, n=5;
6      for (i=0; i<n; i++)
7         printf("y[%d]=%d\n", i,y[i]);
8   }
```

Arrays and `for`-loop are very good friends!

4. Type the following program carefully. Compile and run.

Save As: arr2.c (you can add or change highlighted lines in arr.c then save as arr2.c).

```
    123456789012345678901234567890123456789012345678901234567890123456789012
1   #include <stdio.h>
2   main()
3   {
4      float x, y[5]; /* define (only) array of 5 floats */
5      int i, n=5;
6      for (i=0; i<n; i++)
7      {
8         scanf("%f", &x);
9         y[i] = x;
10     }
11     for (i=0; i<n; i++)
12        printf("y[%d]=%f\n", i,y[i]);
13  }
```

Extra Work

5. Read the following program. See Textbook "Frequency Arrays" on page 217 and "Histograms" on page 218 for more explanation.

```
   12345678901234567890123456789012345678901234567890123456789012345678901234567890123456789012
1  #include <stdio.h>
2  main()
3  {
4
5     /* x in the range 0-9 */
6     const int n=20;
7     int x[]={8,5,0,8,1,
8              1,2,3,0,0,
9              1,6,7,8,3,
10             4,5,1,0,9};
11
12    /* frequency array: freq[0] to freq[9],
13       where freq[i]= how many times i is repeated in x
14    */
15    const int max=10;
16    int freq[max];
17    int i,j;
18
19    /* initialize frequency array */
20    for (i=0; i<max; i++)
21       freq[i] = 0;
22
23    /* calculate number frequencies in x */
24    for (i=0; i<n; i++)
25       freq[x[i]] ++;
26
27    /* print frequency histogram */
28    for (i=0; i<max; i++)
29    {
30       printf("%d: %d ", i,freq[i]);
31
32       for (j=0; j < freq[i]; j++)  /* histogram length depends on freq[i] */
33          printf("*");
34       printf("\n");
35    }
36
37 }
```

Notice how comments made the program easier to understand. Also notice how the empty lines between program sections make the program easier to read. In line 7, we let the C compiler count how many elements in `x[]`. Be careful that `n` should be less than or equal the number of elements used to initialize `x[]`.

6. Type the program in 5 above carefully and save as freq.c. Compile and run.

Change `n` to 21. Write the error message? What does it mean (in Arabic)?

Error message:_____

Meaning:_____

Homework

1. Write the pseudo code for the algorithm used in freq.c (use the pseudo code in Section 8.3 as example).

Lab 8

Key terms are in **bold**. Review definitions of these terms in Textbook Section 9.5. Review algorithms in Textbook Section 8.6.

- ☑ **Function**s.
- ☑ First sort algorithm: selection sort.
- ☑ Tracing programs.

Student Notes

Lab Sheet

1. Change findLargest2.c as follows. Compile and run.

Save As: findLargest3.c

```
    123456789012345678901234567890123456789012345678901234567890123456789012
 1  #include <stdio.h>
 2  int large(int x1, int x2); /* function declaration */
 3
 4  main()
 5  {
 6     int x, largest=0, i=0, n=5;
 7     for (i=0; i<n; i++)
 8     {
 9        scanf("%d", &x);
10        largest = large(x,largest);  /* function call */
11     }
12     printf("largest=%d\n", largest);
13  }
14
15  /* function definition: large takes 2 numbers, returns the largest of the two */
16  int large(int x1, int x2) /* function header */
17  {
18     int big;
19     if (x1 > x2)
20        big = x1;
21     else
22        big = x2;
23     return big;
24  }
```

2. Type the following program carefully. Compile and run.

Save As: selsort.c

```
    123456789012345678901234567890123456789012345678901234567890123456789012
 1  #include <stdio.h>
 2  main()
 3  {
 4     int x[]={23,78,45,8,32,56}, size=6;
 5     int i, imin, j;
 6     int temp;
 7
 8     for (i=0; i<size-1; i++)
 9     {
10
11        /* search for smallest in unsorted list (element i to array end) */
12        imin = i;
13        for (j=i; j<size; j++)
14           if ( x[j] < x[imin] )
15              imin=j;  /* store position of smallest */
16
17        /* in unsorted list, swap smallest with first */
18        temp = x[i];
```

```
19        x[i] = x[imin];
20        x[imin] = temp;
21     }
22
23     /* show sorted list */
24     for (i=0; i<size; i++)
25        printf("%d ", x[i]);
26  }
```

This program implements the selection sort algorithm (Textbook page 154). **Trace** the program (follow step-by-step) using the following table. Each line contains the values *after* the step.

0≤i≤4	i≤j≤5	imin	temp	x					
				[0]	[1]	[2]	[3]	[4]	[5]
0	0								
	1								
	2								
	3								
	4								
	5								
1	1								
	2								
	3								
	4								
	5								
2	2								
	3								
	4								
	5								
3	3								
	4								
	5								
4	4								
	5								

Extra Work

3. Type the following program carefully. Compile and run.

Save As: selsort2.c

```
    123456789012345678901234567890123456789012345678901234567890123456789012
 1  #include <stdio.h>
 2  void selSort(int x[], int n);  /* function declaration: void means nothing */
 3  main()
 4  {
 5     int x[]={23,78,45,8,32,56}, size=6;
 6     int i;
 7
 8     selSort(x,size);   /* function call */
 9
10   /* show sorted list */
11    for (i=0; i<size; i++)
12       printf("%d ", x[i]);
13  }
14
15  void selSort(int x[], int n)   /* function header */
16  {
17     int i, imin, j;
18     int temp;
19     for (i=0; i<n-1; i++)
20     {
21
22        /* search for smallest in unsorted list (element i to array end) */
23        imin = i;
24        for (j=i; j<n; j++)
25           if ( x[j] < x[imin] )
26              imin=j;  /* store position of smallest */
27
28        /* in unsorted list, swap smallest with first */
29        temp = x[i];
30        x[i] = x[imin];
31        x[imin] = temp;
32     }
33  }
```

Try to convert lines 10 to 12 to a function, `printArray()`. Use the following function header
`void printArray(int x[], int n)`
where `n` is the array size. You must remove `i` from `main()` and declare it within `printArray()`.

Homework

1. In the Textbook (pages 185-186), review the definition of the following terms: **function declaration**, **function definition**, **function call**, **function header**, **function body**, **formal parameters**, **actual parameters**.
2. Give examples for the following from the programs of this lab: formal and actual parameters.
3. Use selsort.c to write the pseudo code for the selection sort (Textbook Exercise 46 on page 165).

Lab 9

Key terms are in **bold**. Review definitions of these terms in Textbook Section 9.5. Review algorithms in Textbook Section 8.6.

- ☑ Sort: bubble sort algorithm.
- ☑ Search.: first search algorithm.
- ☑ More functions.

Student Notes

Bubble sort, no function
first search (linear), no function
convert bubble sort and linear search to functions
exercise: use linear search function with array (call from main)

Lab 10

Key terms are in **bold**. Review definitions of these terms in Textbook Section 9.5.

- ☑ More functions.
- ☑ More sort: sort functions.
- ☑ Search.: binary search function.

Student Notes

Insertion sort function
Binary search function
Use binary search function to search array

Exercises

Additional exercises with answers at the end
Challenging problems for advanced students