Information and Computer Science Department

First Semester 162

ICS 103 – Computer Programming in C

Major 01 Examination Key

Saturday, March 25, 2017 (9:00-11:00AM)

Duration: 120 minutes

|  |  |  |  |
| --- | --- | --- | --- |
| Name: |  | Lect Serial # |  |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ID#: |  |  |  |  |  |  |  |  |  |

Please tick your section:

|  |  |
| --- | --- |
| Instructor | Section |
| Dr. Louai Al-Awami | [ ] 01 (UT 7:00 – 7:50) [ ] 02 (UT 7:00 – 7:50)[ ] 03 (UT 8:00 – 8:50) [ ] 04 (UT 8:00 – 8:50) |
| Dr. Samer Arafat | [ ] 05 (UT 11:00 – 11:50) [ ] 06 (UT 11:00 – 11:50) |
| Dr. Mohammed Balah | [ ] 07 (UT 13:10 – 14:00) [ ] 08 (UT 13:10 – 14:00)  |
| Mr. Said Abdallah Muhammad | [ ] 09 (MW 7:00 – 7:50) [ ] 10 (MW 7:00 – 7:50)  |
| Dr. Tarek El-Basuny | [ ] 11 (MW 8:00 – 8:50) [ ] 12 (MW 8:00 – 8:50) |
| Dr. Rafiul Hassan | [ ] 13 (MW 9:00 – 9:50) [ ] 14 (MW 9:00 – 9:50)[ ] 15 (MW 10:00 – 10:50) [ ] 16 (MW 10:00 – 10:50) |
| Dr. El-Sayed El-Alfy | [ ] 17 (MW 11:00 – 11:50) [ ] 18 (MW 11:00 – 11:50)[ ] 19 (MW 13:10 – 14:00) [ ] 20 (MW 13:10 – 14:00) |

Instructions**:**

1. Answer all questions. Make sure your answers are clear and readable.
2. Make sure there are **7** questions in **13** pages.
3. The exam is closed book and closed notes. No calculators or any helping aides are allowed. Make sure to turn off your mobile phone and keep it in your pocket.
4. If there is no space on the front of a question’s page, use the back of the page. Indicate this clearly.

|  |  |  |  |
| --- | --- | --- | --- |
| Question # | Max Mark | Obtained Mark | Remarks |
| 1 | 15 |  |  |
| 2 | 12 |  |  |
| 3 | 13 |  |  |
| 4 | 18 |  |  |
| 5 | 20 |  |  |
| 6 | 22 |  |  |
| Total | 100 |  |  |

**Question 1 (15 points):** Fill this table by selecting the **most** correct answer for each of the following 10 questions:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Question | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| Answer | C | D | E | C | A | B | A | C | D | E |

1. Which step of Software Development involves writing the source code?
2. Analysis
3. Design
4. Implementation
5. Testing
6. Maintenance
7. Which of the followings are valid identifiers in the C language?

i. ICS-103 ii. C\_prog iii. 2nd\_exam

 iv. switch v. @kfupm vi. circle area

1. i, ii, iv, v
2. i, ii,vi
3. ii, iii
4. ii
5. ii,v,vi
6. Which of the following are **not** reserved words in the C language:

i. int ii. return iii. name

iv. sqrt v. else

1. i, iii
2. ii, iv, v
3. iii only
4. ii, v
5. iii, iv
6. What is returned by the following function call?

abs(-3.1415)

1. 3.1415
2. 3.14
3. 3
4. 3.0
5. None of the above.
6. The expression shown below is equivalent to which answer?

!(A < 5 && !(B > 30))

1. A >= 5 || B > 30
2. A <= 5 || B <= 30
3. A > 5 || B > 30
4. A < 5 && B <= 30
5. A >= 5 && B > 30
6. What is displayed on the screen after executing the following statement?

printf("%d %d", 'A' > 'B', 50 % 4);

* 1. 1 0
	2. 0 2
	3. 1 3
	4. 0 1
	5. 1 1
1. Assume that **x** is variable of type **double** that is initialized with a positive value and **y** is an integer variable. The correct way to round off the value of **x** to the nearest integer is:
	1. y = (int)(x + 0.5);
	2. y = int(x + 0.5);
	3. y = (int)x + 0.5;
	4. y = (int)((int)x + 0.5);
	5. y = (int)x;

For example if x =7.6, y will be assigned the rounded value 8 If x=7.3, y will be assigned the rounded value 7

1. Given the following declarations:

 double a = 0.0112; int b = -23;

Which print statement produces the following output? (*A square represents one space*)

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 0 | . | 0 | 1 |  | - | 2 | 3 |  | 1 | . | 1 | 2 | e | - | 0 | 0 | 2 |

* 1. printf("%5.2f%3d%11.2e", a, b, a);
	2. printf("%6.2f%3d%9.2e", a, b, a);
	3. printf("%5.2f%4d%10.2e", a, b, a);
	4. printf("%2.3f%2d%1.2e", a, b, a);
	5. printf("%5.3f%4d%9.2e", a, b, a);
1. What is the output of the following code fragment?

 int k = 4, z = 12;

 if(k = 5 || z > 50);

 printf("Hello");

 printf("Welcome");

* + 1. Hello
		2. Welcome
		3. No output
		4. HelloWelcome
		5. No output. The fragment will cause compilation error
1. Consider the following C program.

#include <stdio.h>

int main(void) {

int k;

scanf("%d",&k);

if(k > 10 || k <= 99)

 printf("%d",(k - k/10\*10)\*10+k/10);

else

 printf("OUT OF RANGE");

return 0;

}

what is the output if the input is 9?

* 1. 0.9
	2. 900
	3. OUT OF RANGE
	4. 9
	5. 90

**Question 2 (12 points):** Find the value of each of the following C expressions:

|  |  |
| --- | --- |
| **Expression** | **Value** |
| !6||3&&4 | **1 (true)** |
| 3+5/2.0 | **5.5** |
| 6/4\*4 | **4** |
| 1==!1>1 | **0 (false)** |
| 1+0>=5<1 | **1 (true)** |
| (double)5/(int)2.5 | **2.5** |
| 397%100/10 | **9** |
| !(10!=7) | **0 (false)** |

**Question 3 (13 points):** What is the output of each of the following code fragments?

|  |  |
| --- | --- |
| Code | Output |
|  3 pts int score=10; switch(score){  case 10: printf("Good\n"); case 5: printf("OK"); break; default: printf("BAD"); }  | GoodOK |
| 3 pts  int A=30;  if(A > 10) if(A%3) printf("HOW "); else printf("ARE "); printf("YOU?"); | ARE YOU? |
|  3 pts int x=7;  if(x >= 5 )  if(x > 10) if(x <15) printf("A"); else printf("C"); else if ( x >= 0) printf("D"); else printf("E"); else printf("F"); | D |
|  4 pts int A = 0, B = 10;  if (A) printf("A\n"); if (B) printf("B\n"); if (A = B)  printf("A=B\n"); else  printf("A!=B\n"); | BA=B |

**Question 4 (18 points):** Write a C program for a simple calculator that reads a mathematical expression of the form :

Operand1 operator Operand2

that has two operands and an operator from the set {+ , - , \* , / , ^}. The program then detects the required mathematical operation and computes the result.

The following tables explains what each operator means:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Operator | + | - | \* | / | ^ |
| Operation | Addition | Subtraction | Multiplication | Division | Power |
| Example | 1+1=2.0 | 5-10=-5.0 | 4\*3.5=14.0 | 100/2=50.0 | 2^3=8.0 |

NOTES:

* + - Your program must show a proper prompt to the user informing him on what to enter.
		- You must use **switch** statement to choose the correct mathematical operation.
		- If the operator entered is not one of the five allowed operators, a proper error message must be displayed.

Sample program runs:









#include<stdio.h>

#include<math.h>

int main(void){

 double operand1, operand2; //For storing operands

 char operator; //For storing operation

 //Show prompt and read input

 printf("Enter your expression: ");

 scanf("%lf%c%lf",&operand1,&operator,&operand2);

 //Examin input and choose correct computation

 switch(operator){

 case '+': //Addition

 printf("The Answer is %0.2f",operand1 + operand2);

 break;

 case '-'://Subtraction

 printf("The Answer is %0.2f",operand1 - operand2);

 break;

 case '\*'://Multiplication

 printf("The Answer is %0.2f",operand1 \* operand2);

 break;

 case '/'://Division

 printf("The Answer is %0.2f",operand1 / operand2);

 break;

 case '^'://Power

 printf("The Answer is %0.2f",pow(operand1,operand2));

 break;

 default://Invalid operation

 printf("Invalid Operator");

 }

 return 0;

}

**Question 5 (20 points):** An Armstrong Number is an integer such that the sum of the cube of its digits equals the number itself. For example, 153 is an Armstrong number because 13+53+33= 153.

Write a program that prompts for and reads a 3-digit integer and outputs whether it is an Armstrong number or not. Your program must work for any 3-digit number between 100 and 999. If the input is outside the above range, the program displays a proper error message and terminates.

Sample program runs:

Enter a number between 100 and 999:  **370**

370 is an Armstrong Number.

Enter a number between 100 and 999: **8933**

Error: Invalid input.

Enter a number between 100 and 999: **100**

100 is not an Armstrong Number.

#include<stdio.h>

#include<math.h>

int main(void){

 int H,T,O; //to store hundreds,tens, and ones

 int num; //to store input

 int temp; //temporary variable

 double sum; //to store the sum of digits cube

 //Show prompt and read input

 printf("Enter a number between 100 and 999: ");

 scanf("%d",&num);

 if(num >= 100 && num <=999){

 H = num/100; //isolate the hundreds digit

 temp = num%100; //isolate the tens and the ones digits

 T = temp/10; //isolate the tens digit

 O = temp%10; //isolate the ones digit

 sum = pow(H,3)+pow(T,3)+pow(O,3); //sum of cubes

 if(sum == num) //Yes, an Armstrong Number

 printf("%d is an Armstrong Number.",num);

 else //NO, Not an Armstrong Number

 printf("%d is NOT an Armstrong Number.",num);

 }

 else{ // Input is invalid

 printf("Error: Invalid input.");

 }

 return 0;

}

**Question 6 (22 points):** A projectile is launched at an angle theta, θ, to the horizontal with an initial velocity$ v\_{0}$. It rises upwards to a maximum height while moving horizontally with a velocity$ v\_{h}$. Write a C program that prompts for and reads the initial velocity $v\_{0} $in m/s and the angle theta in degrees. Based on this input, the program computes and displays the maximum height, in meters, reached by the projectile and the time taken to reach this height.

Note that the maximum height is reached when the vertical velocity$ V\_{v}$ is 0. The equations of the height and vertical velocity are given below:

$height=v\_{0}\sin(\left(θ\right))t- 0.5gt^{2}$ $V\_{v}=v\_{0}\sin(\left(θ\right))-gt$

Where:

$g$ is the gravity acceleration to be defined as a constant with a value of **9.8** m/s2

$$180 degrees= π radians$$

$π $ is to be defined as a constant with a value of **3.14159**

 

To solve this problem, your program must contain, in addition to the main function, 3 other functions:

**degToRad** function that receives the angle in degrees and returns it in radians.

**getTimeToPeak** function that receives the initial velocity and the angle theta in radians and it returns the time to reach the maximum height in seconds.

**getMaxHeight** function that receives the initial velocity , the angle theta in radians, and the time to reach maximum height in seconds and returns the maximum height in meters.

The range of the valid input is:

 0$\leq θ\leq 90 $ degrees $v\_{0} \geq 0$

If the input is not valid, your program displays “Error: Wrong input” message and terminates.

**Note:**

* You must put the definition of the **main** function before those of **degToRad**, **getTimeToPeak**, and **getMaxHeight**.
* The functions **degToRad**, **getTimeToPeak**, and **getMaxHeight** must not contain any **printf** or **scanf** calls. All reading and printing must be done in the **main** function.

Sample program runs:

|  |  |  |
| --- | --- | --- |
|  | Enter v0 in m/s: -34.5Enter theta in degrees: 40.5Error: Wrong input | Enter v0 in m/s: 30.8Enter theta in degrees: 120.6Error: Wrong input |

#include <stdio.h>

#include <math.h>

#define G 9.8

#define PI 3.14159

double degToRad(double degrees);

double getTimeToPeak(double v0, double theta);

double getMaxHeight(double time, double v0,double theta);

 int main(void){

 double v0, thetDeg, thetRad,peakTime, maxHeight;

 printf("Enter v0 in m/s: ");

 scanf("%lf",&v0);

 printf("Enter theta in degrees: ");

 scanf("%lf",&thetDeg);

 if(v0 < 0 || thetDeg > 90 || thetDeg <0 )

 printf("Error: Wrong input\n");

 else {

 thetRad=degToRad(thetDeg);

 peakTime=getTimeToPeak(v0,thetRad);

 maxHeight= getMaxHeight(peakTime,v0,thetRad);

 printf("Time to reach max height=%.2f secs\n",peakTime);

 printf("Maximum height = %.2f m\n",maxHeight);

 }

 return 0;

 }

double degToRad(double degrees){

 double rad;

 rad= degrees/180\*PI;

 return rad;

}

double getTimeToPeak(double v0, double theta){

 double time;

 time=v0\*sin(theta)/G;

 return time;

}

double getMaxHeight(double time, double v0, double theta){

 double height;

 height=v0\*sin(theta)\*time -0.5\*G\*time\*time;

 return height;

}