****

**INF211**

**ALGORITHMS AND PROGRAMMING**

**PROJECT-3**

**PROJECT-3:** **Mathematical operation, variables and printf/scanf commands in C**

**Deadline is November 1, 2020 at 23:00.**

**Projects that are not delivered on time are not accepted.**

**Upload the project to the Project 3 assignment section of the INF211 class.**

**The questions can be asked to course lecturer Dr. Tuba Gözel and teaching assistant Asıhan Aktepe under the Project 3 announcement of INF211 class Week3 channel.**

**The questions should be answered in this document. This document and the C code files for the questions from #9 to #12 should be uploaded to the assignment section.**

**The code filenames should be formatted as ID\_N\_P3\_Q#.C**

***Note:***

* ***Do not forget to write the meaning of what you do in the code as a comment.***
* ***You must also send the code files.***
* ***Code Filename should include school ID, the first letter of student’s whole name (registered name in the student information system), Project Number and Question Number.***

***Filename format should be* ID\_FirstLetterofName\_P#\_Q#**

***Example: The code filename of Veli Zeki Doğan for question 10 should be* 201029999\_VZD\_P3\_Q10.C**

***Questions:***

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1. **(1-Point) Indicate which of the following are valid type int , double , or char constants in C and which are not. Identify the data type of each valid constant.**  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | 'PQR' | 15E-2 | 'h' | -37.491 | .912 | 4,719 | 'true' | "T" | 4.5e3 | '$' | | invalid | double | char | integer | Invalid double | Invalid double | invalid | invalid | double | char |  1. **(1-Point) What would be the best variable type for the area of a circle in square inches? Which type for the number of cars passing through an intersection in an hour? The first letter of your last name?**  |  | | --- | | **double** | | **integer** | | **char** |  1. **(2-Point) Given the constants and variable declarations**   #define PI 3.14159  #define MAX\_I 1000  . . .  double x, y;  int a, b, i;  **indicate which of the following statements are valid, and find the value stored by each valid statement. Also indicate which are invalid and why. Assume that**  a is 3, b is 4, and y is −1.0.   |  |  | | --- | --- | | i = a % b | 3 | | i = (989 −MAX\_I) / a | -3 | | i = b % a | 1 | | x = PI \* y | -3.14159 | | i = a / −b | 0 | | x = a / b | 0.000000 | | x = a % (a / b) | Invalid because a/b is zero and dividing a number by zero is indefinite. | | i = b / 0 | Invalid because dividing a number by zero is indefinite. | | i=a%(990−MAX\_I) | 3 | | i = (MAX\_I−990)/a | 3 | | x = a / y | -3.000000 | | i = PI \* a | 9 | | x = PI / y | -3.141590 | | i = a % 0 | Invalid because dividing a number by zero is indefinite. | | x = (double) a / b | 0.750000 |  1. **(1-Point) Assume that you have the following variable declarations:**   int color, lime, straw, red, orange;  double white, green, blue, purple, crayon;    **Evaluate each of the statements below using the following values:**  color is 2, crayon is −1.3 , straw is 1 , red is 3 , purple is 0.3E+1 .   |  |  | | --- | --- | | white = color \* 2.5 / purple | 1.666667 | | green = color / purple | 0.666667 | | orange = color / red | 0 | | blue = (color + straw) / (crayon + 0.3) | -3.000000 | | lime = red / color + red % color | 2 | | purple = straw / red \* color | 0.000000 |  1. **(1-Point) Evaluate the following, assuming that letters have consecutive character codes.**  |  |  | | --- | --- | | (int)'D' − (int)'A' | **3** | | (char)((int)'C' + 2) | **E** | | (int)'6' − (int)'7' | **-1** |  1. **(1-Point) Evaluate the following expressions if** x is 10.5, y is 7.2, m is 5, and n is 2**.**  |  |  | | --- | --- | | x / (double)m | **2.10** | | x / m | **2.10** | | (double)(n \* m) | **10.00** | | (double)(n / m) + y | **7.20** | | (double)(n / m) | **0.00** |  1. **(1-Point) Write an assignment statement that might be used to implement the following equation in C.**  |  |  | | --- | --- | |  | **q=(k\*A\*(T1-T2))/L** |  1. **(2-Point) What is the result difference between the following expressions? (i=5)**  |  |  | | --- | --- | | i++ | **i=5** (if we print this out, it’s going to display “i” first then “i” is going to increased 6.) | | ++i | **i=6** (if we print this out, “i” is going to increased 6 first then it’s going to print “i” out. | | j=i++ | **j = 5 and i=6** (predence of “=” is higher than postfix “++” so, value of “i” is assigned to “j“ before incrementing) | | j=++i | **j = 6 and i=5** (precedence of prefix “++” is more than “=” operator. So “i” will increment first and the incremented value is assigned to “j”) |  1. **(18-Point) Please write the code that calculates and prints the results of:**   **a+c, t+c, y+t, a+t, s+b, w+b, s+c, w+c, w+pp.**  **Take: a=200, b=12345, w=1234567890, s = 4043, t = 2.13459, y = 1.1415927**  **Output should be:**  *a + c = 275*  *t + c = 77.134590*  *y + t = 3.276183*  *a + t = 202.134583*  *s + b = 16388*  *w + b = 1234580235*  *s + c = 4118*  *w + c = 1234567965*  *w + pp =3776135780*   |  | | --- | | **\*write your code here and upload its file( XXXXXX\_XXXX\_P3\_Q9.C)**  **#include <stdio.h>**  **int main()**  **{**  **int a=200,b=12345,w=1234567890, s=4043;**  **int c=275-a;**  **unsigned int pp=3776135780-w;**  **double t=2.13459, y=1.1415927;**  **printf("a+c = %d\n", a+c);**  **printf("t+c = %f\n", t+c);**  **printf("y+t = %f\n", y+t);**  **printf("a+b = %d\n", a+b);**  **printf("s+b = %d\n", s+b);**  **printf("w+b = %d\n", w+b);**  **printf("s+c = %d\n", s+c);**  **printf("w+c = %d\n", w+c);**  **printf("w+pp = %u\n", w+pp);**  **return 0;**  **}** |  1. **(18-Point) Write a C programming to display the sizes for each of C's data types.**   **Output should be:**  *Size of C data types*  *Type Bytes*  *char 1*  *unsigned char 1*  *short 2*  *int 4*  *unsigned 4*  *long 4*  *unsigned long 4*  *long long 8*  *unsigned long long 8*  *float 4*  *double 8*  *long double 16*   |  | | --- | | **\*write your code here and upload its file( XXXXXX\_XXXX\_P3\_Q10.C)**  **#include <stdio.h>**  **int main()**  **{**  **printf(" Size of c data types\n");**  **puts(" Type Bytes");**  **printf("\n char 1");**  **printf("\n unsigned char 1");**  **printf("\n short 2");**  **printf("\n int 4");**  **printf("\n unsigned 4");**  **printf("\n long 4");**  **printf("\n unsigned long 4");**  **printf("\n long long 8");**  **printf("\n unsigned long long 8");**  **printf("\n float 4");**  **printf("\n double 8");**  **printf("\n long double 16\n");**  **return 0;**  **}** |  1. **(18-Point) Write a program that calculates mileage reimbursement for a salesperson at a rate of $0.35 per mile. Your program should interact with the user in this manner:**   MILEAGE REIMBURSEMENT CALCULATOR  Enter beginning odometer reading=> **13505.2**  Enter ending odometer reading=> **13810.6**  You traveled 305.4 miles. At $0.35 per mile,  your reimbursement is $106.89.   |  | | --- | | **\*write your code here and upload its file( XXXXXX\_XXXX\_P3\_Q11.C)**  **#include <stdio.h>**  **int main()**  **{**  **double beginning,ending;**  **float miles,reimbursement,price=0.35;**  **puts(“MILAGE REIMBURSEMENT CALCULATOR\n”);**  **printf("Enter beginning odometer reading: ");**  **scanf("%lf",&beginning);**  **printf("\nEnter ending odometer reading: ");**  **scanf("%lf",&ending);**  **miles= ending- beginning;**  **if (ending>=beginning){**  **printf("\nYou have traveled %.2f miles.At $0.35 per mile", miles);**  **reimbursement= miles\*price;**  **printf("\nYour reimbursement is: %.2f \n", reimbursement);**  **}**  **else{**  **printf("\nYou should check your odometer again");**  **}**  **return 0;**  **}** |  1. **(18-Point) Write a program to assist in the design of a hydroelectric dam.** Prompt **the user for** the height of the dam **and for** the number of cubic meters of water **that are projected to flow from the top to the bottom of the dam each second. Predict how many megawatts (1MW = 106 W) of power will be produced if 90% of the work done on the water by gravity is converted to electrical energy. Note that the mass of one cubic meter of water is 1000 kg. Use 9.80 meters/second2 as the gravitational constant g. Be sure to use meaningful names for both** the gravitational constant **and the 90%** efficiency constant**. For one run, use a height of 170 m and flow of 1.30x10 3m 3/s. The relevant formula ( w = work, m = mass, g = gravity, h = height) is: w = mgh.**  |  | | --- | | **\*write your code here and upload its file( XXXXXX\_XXXX\_P3\_Q12.C)**  **#include <stdio.h>**  **int main()**  **{**  **double height, water, work, mass;**  **float gravity=9.80, efficiency;**  **puts("Hydroelectric Dam Energy Calculator");**  **printf("\nEnter the height of dam in meters: ");**  **scanf("%lf", &height);**  **printf("\nEnter the number of cubic meters: ");**  **scanf("%lf", &water);**  **mass= water\*1000;**  **efficiency = 0.9;**  **work= mass\*gravity\*height\*efficiency;**  **work = work/1e6;**  **printf("\nThe net energy obtained is %f MW\n", work);**  **return 0;**  **}** |  1. **(18-Point) Explain the flowchart below.**     **İlk önce kullanacağımız tüm değerleri yazıyor. Koordinat düzlemindeki fonksiyonun x üzerindeki iz düşümündeki en küçük değer a, en büyük değer ise c'dir. Bu fonksiyonun sıfır olduğu değer yani kökü olan en az 1 değer vardır. Bu kökü bulmak için ilk önce ikiye aralık toplanıp ikiye bölünür. Bulunan b değeri ile a değerini çarpılıp koşulun sağlanıp sağlanmadığı kontrol edilir. Eğer sonuç negatif çıkarsa kök, a ve b arasında bir yerdedir. Bu yüzden aralığı daraltmak için b değeri c'ye atanır. Eğer negatif çıkmazsa kök, b ile c arasında bir yerde olur. Bu yüzden b değerini a'ya atanır. a ve c'nin uzaklıkları arasındaki fark azaldıkça köke yaklaşılır. Eğer a ve c'nin arasındaki uzaklık fark hata oranından küçük değilse aralığın uç değerleri toplanıp ikiye bölünmek üzere üçüncü adıma geri dönülür. Böylelikle aralık köke en yakın olduğu zamana dek işlem tekrar eder. Küçük olduğu vakit kök değeri b’ye atanır ve işlem biter.** |

**ABOUT THE PROJECT**

This project file is prepared for the INF211 Algorithms and Programming I given in the Department of Electronic Engineering of Gebze Technical University.

|  |  |
| --- | --- |
| **Process** | **Prepared By** |
| Date: 20.10.2020 | Aslıhan AKTEPE |