Bismillahir Rahmanir Rahim

International Islamic University Chittagong

Department of Computer Science & Engineering

Mid Term Examination, Autumn 2022

CSE-1121 Computer Programming I

[Solution]

1(a)

Name	**Format Specifier**	**Size (bytes)**	**Minimum Value**	**Maximum Value**
int	%d	at least 2, usually 4	-2^{31}	$2^{31} - 1$
char	%c	1	-2^{7}	$2^{7} - 1$
float	%f	4	1.2×10^{-38}	3.4×10^{38}
double	%lf	8	1.7×10^{-308}	1.7×10^{308}

1(b)

$p = 10^5$, $q = 10^5$. Hence, result = 10^{10} which is greater than $2^{31} – 1$. An integer variable can store values not greater that $2^{31} – 1$. So, the result of the expression 'p * q' will be overflowed, resulting in inability of storing the value correctly.

One of the ways to make the correction could be:

```
int main()
{
      long long p = 100000;
      long long q = 100000;
      long long result = p * q;
      printf("%lld", result);
      return 0;
}
```

1(c)

The output will be:

Not Equal

Reason: In C language, floating point numbers can't be stored accurately since the numbers aren't stored in binary format.

1(d)

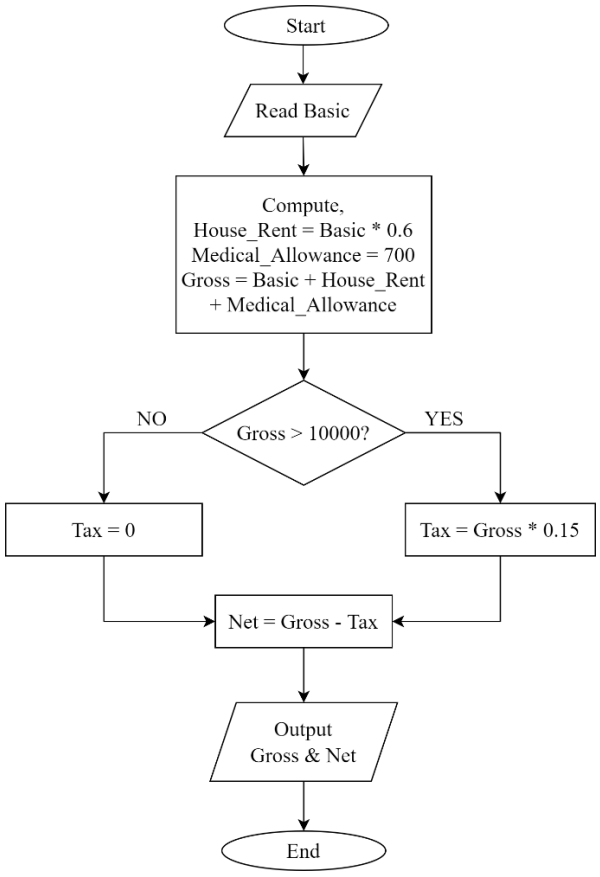
The code won't produce correct output. In C language, the arithmetic division operation between two integer operands will produce an integer quotient. Hence, 5 / 9 will be 0, thus 'cel' will always be 0 irrespective of the value of 'F'.

One of the ways to make the correction could be:

```
cel = (5 / 9.0) * (F - 32);
```

1(e)

Flowchart:



Algorithm:

Step 1: Start the program

Step 2: Read Basic from user

Step 3: Compute,

House_Rent = Basic * 0.6

Medical_Allowance = 700

Gross = Basic + House_Rent + Medical_Allowance

Step 4: If Gross > 10000, Tax = Gross * 0.15

Else, Tax = 0

Step 5: Compute, Net = Gross – Tax

Step 6: Output Gross & Net pay

Step 7: End of the program

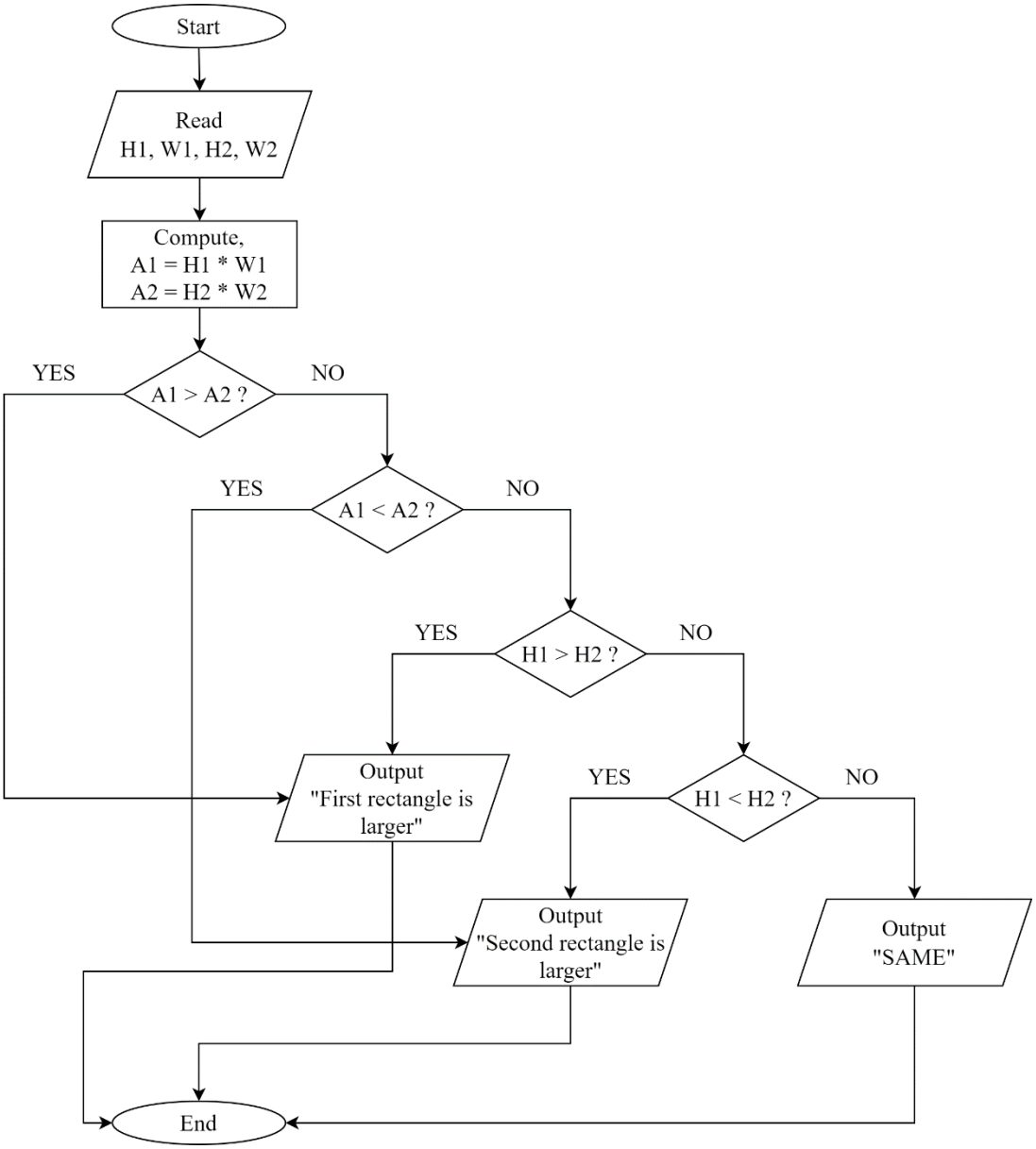
Source Code:

```
#include <stdio.h>

int main()
{
	double Basic, House_Rent, Medical_Allowance, Gross, Net, Tax;
	scanf(“%lf”, &Basic);

	House_Rent = Basic * 0.6;
	Medical_Allowance = 700;
	Gross = Basic + House_Rent + Medical_Allowance;

	if (Gross > 10000)
		Tax = Gross * 0.15;
	else
```

```
        Tax = 0;
    Net = Gross – Tax;
    printf("Gross Pay = %.2lf\nNet Pay = %.2lf\n", Gross, Net);
    return 0;
}
```

1(e) [OR]

Flowchart:



Algorithm:

Step 1: Start the program

Step 2: Read H1, W1, H2, W2 from user

Step 3: Compute,

A1 = H1 * W1,

A2 = H2 * W2

Step 4: If A1 > A2, Go to Step 8.

Else, Go to Step 5.

Step 5: If A1 < A2, Go to Step 9.

Else, Go to Step 6.

Step 6: If H1 > H2, Go to Step 8.

Else, Go to Step 7.

Step 7: If H1 < H2, Go to Step 9.

Else, Go to Step 10.

Step 8: Print “First rectangle is larger”, Go to Step 11.

Step 9: Print “Second rectangle is larger”, Go to Step 11.

Step 10: Print “SAME”, Go to Step 11.

Step 11: End of the program.

Source Code:

```
#include <stdio.h>

int main()
{
      int H1, W1, H2, W2, A1, A2;
      scanf(“%d%d%d%d”, &H1, &W1, &H2, &W2);

      A1 = H1 * W1;
```

```
    A2 = H2 * W2;
    if (A1 > A2)          printf("First rectangle is larger\n");
    else if (A1 < A2)     printf("Second rectangle is larger\n");
    else if (H1 > H2)     printf("First rectangle is larger\n");
    else if (H1 < H2)     printf("Second rectangle is larger\n");
    else             printf("SAME\n");
    return 0;
}
```

2(a)

i) k = 0
ii) k = 3, z = 3.000000
iii) y = 5.90000
iv) i = 29, j = 15, k = 29

2(b)

i)

0	0	0	0	8	1	2	3

ii)

3	4	.	5	7					

iii)

				Q	u	a

2(c)

i) 8
ii) 6.0
iii) 10.0
iv) 65536.0
v) 0 (false)
vi) 8

2(d)

```
#include <stdio.h>

int main()
{
      int held, attended;
      double perc;
      scanf(“%d%d”, &held, &attended);
      perc = (attended * 100.0) / held;
      if (perc < 70.0) printf(“%.2lf%%, Not Allowed\n”, perc);
      else             printf(“%.2lf%%, Allowed\n”, perc);
      return 0;
}
```

2(d) [OR]

```
#include <stdio.h>

int main()
{
      int x1, y1, x2, y2;
      scanf(“%d%d%d%d”, &x1, &y1, &x2, &y2);
      if ((x1 >= 0 && x2 >= 0 || x1 < 0 && x2 < 0) && (y1 >= 0 && y2 >=
0 || y1 < 0 && y2 < 0))
            printf(“Yes\n”);
      else
            printf(“No\n”);
      return 0;
}
```

3(a)

i) Output will be “1 -8”. Since “n == 10” condition is false and the else is connected to the later if-block, “y -= 10” expression is evaluated.

ii) Output will be “1 2”. Since “n > 5” condition is false and there is no else-if or else block connected to the first if-block, no expression is evaluated.

3(b)

```
switch (color)
{
      case ‘R’:
      case ‘r’:
            printf(“Red\n”);
            break;
      case ‘G’:
      case ‘g’:
            printf(“Green\n”);
            break;
      case ‘B’:
      case ‘b’:
            printf(“Blue\n”);
            break;
      default:
            printf(“Not a prime color\n”);
}
```

3(c)

Using for-loop:

```
int i;
for (i = 100; i >= 1; i -= 2)
{
      printf(“%d\n”, i);
}
```

Using while-loop:

```
int i = 100;
while (i >= 1)
{
      printf(“%d\n”, i);
      i -= 2;
}
```

3(d)

```
#include <stdio.h>

int main()
{
      int x, i;
      scanf(“%d”, &x);
      printf(“1”);
      for (i = 3; i < x; i += 2)
      {
            if (x % i == 0) printf(“ %d”, i);
      }
```

```
	return 0;
}
```

3(d) [OR]

```
#include <stdio.h>

int main()
{
	int n, m, ones = 0;
	printf(“Enter a positive integer: ”);
	scanf(“%d”, &n);
	m = n;
	while (m != 0)
	{
		ones += m % 2;
		m /= 2;
	}
	if (ones % 2 == 0)    printf(“%d is an Evil number.\n”, n);
	else             printf(“%d is an Odious number.\n”, n);
	return 0;
}
```