1. Given the array [3,5,4,9,2], sort this array using the Selection sort. Show all of your working with annotation at each step to show what you are doing.

1. Given the array [3,5,4,9,2], sort this array using the Selection sort. Show all of your working with annotation at each step to show what you are doing.

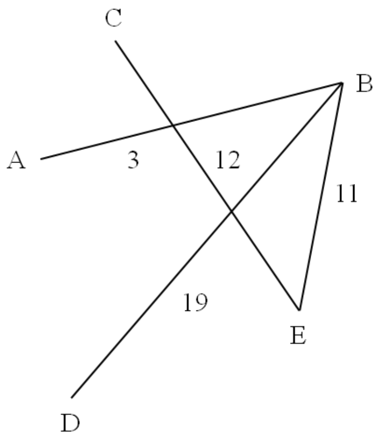
```
[3][5][4][9][2]    original array
3 is the initial lowest element
[3][5][4][9][2]    compare 3 to 5, 3 is still smallest
[3][5][4][9][2]    compare 3 to 4, 3 is still smallest
[3][5][4][9][2]    compare 3 to 9, 3 is still smallest
[3][5][4][9][2]    compare 3 to 2, 2 is now the smallest
2 is the smallest element
[2][5][4][9][3]    swap 3 and 2

5 is the initial smallest element
[2][5][4][9][3]    compare 5 to 4, 4 is now the smallest
[2][5][4][9][3]    compare 4 to 9, 4 is still the smallest
[2][5][4][9][3]    compare 4 to 3, 3 is now the smallest
3 is the smallest element
[2][3][4][9][5]    swap 5 and 3

4 is the initial smallest element
[2][3][4][9][5]    compare 4 to 9, 4 is still the smallest
[2][3][4][9][5]    compare 4 to 5, 4 is still the smallest
4 is the smallest element
[2][3][4][9][5]    swap 4 and 4 (redundant)

9 is the initial smallest element
[2][3][4][9][5]    compare 9 to 5, 5 is now the smallest
5 is the smallest element
[2][3][4][5][9]    swap 9 and 5

9 is the smallest element
[2][3][4][5][9]    swap 9 and 9 (redundant) we are sorted!
```

4. Given the array [“Gill”,”Ron”,”Eva”,”Ali”,”Tom”], sort this array using the Insertion sort. Show all your working with annotation at each step to show what you are doing.

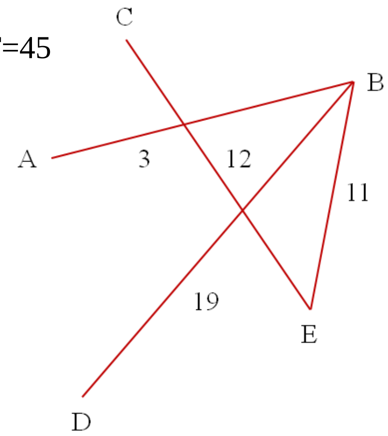
```
[“Gill”][“Ron”][“Eva”][“Ali”][“Tom”]  original array

Start from “Ron”, as “Ron”>Gill” insert in position 1 (redundant)
[“Gill”][“Ron”][“Eva”][“Ali”][“Tom”]  after inserting “Ron”

Start from “Eva”, as “Eva”<”Ron” move “Ron” to position 2, as
“Eva”<”Gill” move “Gill” to position 1, insert “Eva” in position 0
[“Eva”][“Gill”][“Ron”][“Ali”][“Tom”]  after inserting “Eva”


Start from “Ali”, as “Ali”<”Ron” move “Ron” to position 3, as
“Ali”<”Gill” move “Gill” to position 2, as “Ali”<”Eva” move “Eva”
to position 1, insert “Ali” in position 0
[“Ali”] [“Eva”][“Gill”][“Ron”][“Tom”] after inserting “Ali”
we are sorted!
```

2. Determine the minimum spanning tree in each case below. Show all your working in each case.

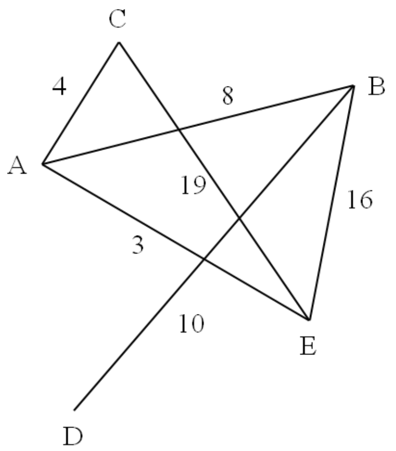
a.



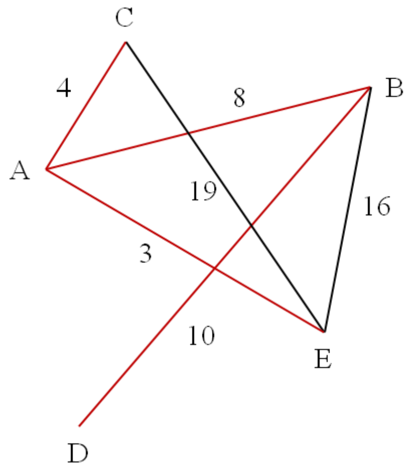
MST=45



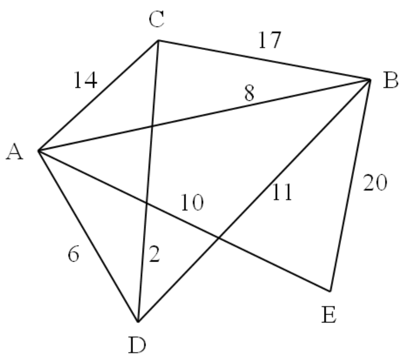
b.

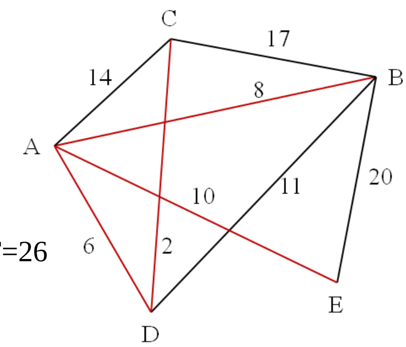


MST=25



c.





MST=26

1. Integrate the following analytically (by hand):
    a. $\int_0^1 x^3\,dx$
    [x^4/4] = 1/4 - 0 = 1/4

    b. $\int_{0.5}^5 2x^3 - 5x\,dx$
    [x^4/2-5x^2/2] = 250 – (-0.59375) = 250.59375
    c. $\int_2^4 x^5 + 2x^2 + 5x^4\,dx$
    [x^6/6+2x^3/3+x^5] = 1749.3333-48 = 1701.33

    d. $\int_{-2}^1 205x^{12}\,dx$
    [205x^13/13] = 15.76923-(-129181.53846) = 129197.30769
    e. $\int_{-2}^{-2} 205x^{12}\,dx$ (explain your answer)
    0
    f. $\int_{1.5}^3 e^{2x}\,dx$
    [1/2e^2x] = 201.7144-10.042768 = 191.67163

2. Use the Trapezium rule with 4 strips (n=4) to work out by hand the area under the curve of
$f(x) = 200x^2 + 65x^3 + 20x^4 + 15x^5$

from a lower limit *a*=0.5 to an upper limit *b*=1.1 .

Work out the analytical (exact answer) and then calculate the absolute true error for the numerical scheme.

Trapezoidal rule

i	xi	f(xi)	2*f(xi)		
0.000000000	0.500000000	59.843750000			
1.000000000	0.650000000	107.661185938	215.322371875		
2.000000000	0.800000000	174.387200000	348.774400000		
3.000000000	0.950000000	264.126214063	528.252428125	solution=	115.061070000
4.000000000	1.100000000	381.954650000		abs true err=	1.178190000
				abs rel true err=	0.010345629

3. Use Simpsons Rule with 6 strips (n=6) to calculate by hand

$$f(x) = 10x^2 - 6x^3 - 90x^4 + 400x^5$$

from a lower limit a=1 to an upper limit b=3 .

Work out the analytical (exact answer) and then calculate the absolute relative true error for this numerical scheme.

Simpsons rule

2n= 6.000000000

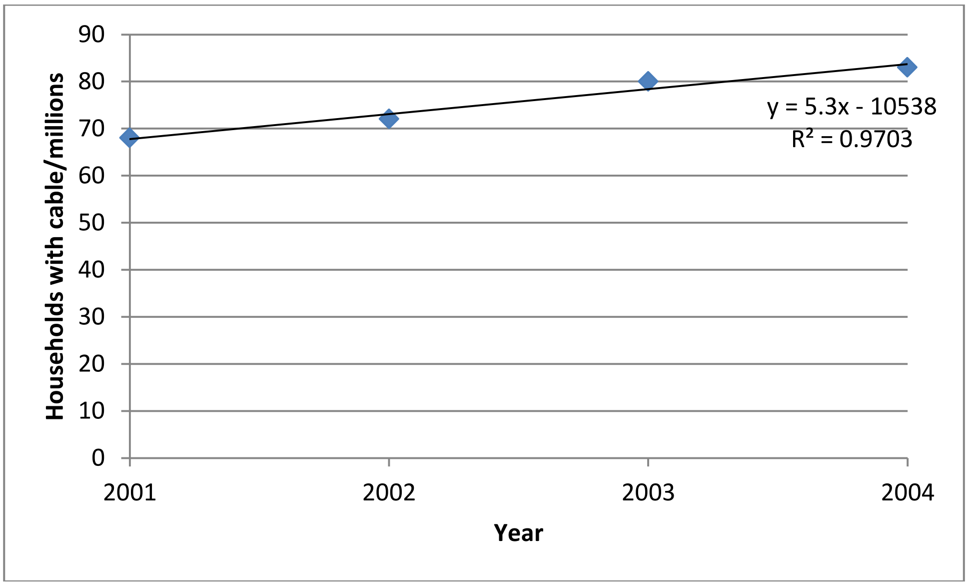
i	xi	f(xi)	4*f(xi)	2*f(xi)		
0.000000000	1.000000000	314.000000000				
1.000000000	1.333333333	1404.707818930	5618.831275720			
2.000000000	1.666666667	4449.588477366		8899.176954733		
3.000000000	2.000000000	11352.000000000	45408.000000000		solution=	44156.872427984
4.000000000	2.333333333	24976.288065844		49952.576131687	abs true err=	12.872427984
5.000000000	2.666666667	49345.316872428	197381.267489712		abs rel true err=	0.000291601
6.000000000	3.000000000	89838.000000000				

3. Some data appeared in a newspaper a few years ago showing the change in the use of Cable TV in households in China.

Year (x)	2001	2002	2003	2004
Households with cable/millions (y)	68	72	80	83

a. Draw a scatter diagram for these data and comment on any noticeable pattern.
b. Obtain by hand, the equation of the least squares regression line of y on x.
c. Interpret the regression coefficients.
d. Plot the regression line on your scatter diagram and comment about its suitability.
e. What is the expected Cable usage in 2008? Is this plausible?
   $y = 5.3x - 10538$ when $x=2008$ $y=104.4$ million people using Cable TV.
   Yes, but always exercise caution with extrapolation.

						sum	mean
Year (x)	2001	2002	2003	2004		8010	2002.5
Households with cable/m (y)	68	72	80	83		303	75.75
y-mean y	-7.75	-3.75	4.25	7.25			
x-mean x	-1.5	-0.5	0.5	1.5			
(x-mean x)(y-mean y)	11.625	1.875	2.125	10.875		26.5	
(x-mean x)^2	2.25	0.25	0.25	2.25		5	
	b1=	5.3		b0=	-10537.5		



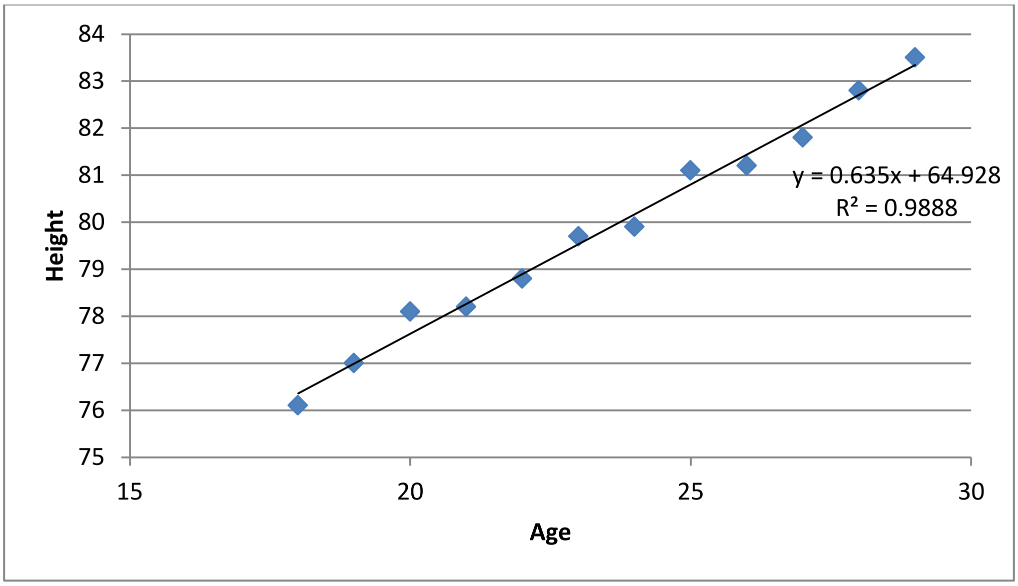
2. An Egyptian village was used as the site of a study of nutrition in developing countries. The data were obtained by measuring the heights (cm) of all 161 children in

the village each month over several years. The data shows the mean heights for each age.

age	height (cm)
18	76.1
19	77
20	78.1
21	78.2
22	78.8
23	79.7
24	79.9
25	81.1
26	81.2
27	81.8
28	82.8
29	83.5

a. Obtain by hand, the equation of the least squares regression line.
b. Plot the regression line and use it to determine the average height for an average 27 year old.
   $y = 0.635x + 64.928$ when $x=27$ $y=82.073$ so average height of 27 yr old is 82.1cms.
c. Calculate the correlation coefficient for this data. What does it tell you about this data set?
   $R^2 = 0.9888$ -? very high degree of correlation so data could be said to follow linear trend. But is it sensible for extrapolation? We do not grow in height every year of our lives?

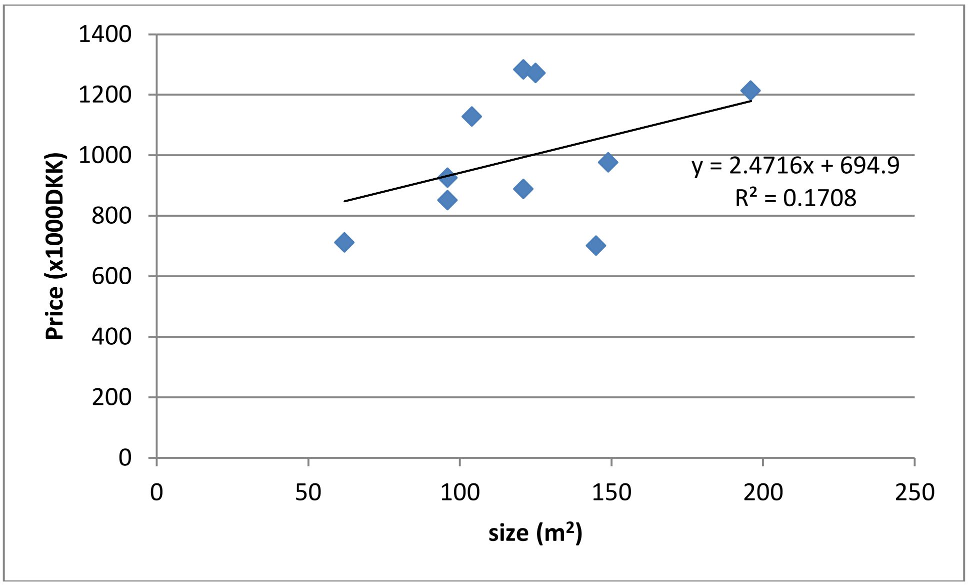
	age (x)	height (y)	y-mean y	x-mean x	(x-mean x)(y-mean y)	(x-mean x)^2
	18	76.1	-3.75	-5.5	20.625	30.25
	19	77	-2.85	-4.5	12.825	20.25
	20	78.1	-1.75	-3.5	6.125	12.25
	21	78.2	-1.65	-2.5	4.125	6.25
	22	78.8	-1.05	-1.5	1.575	2.25
	23	79.7	-0.15	-0.5	0.075	0.25
	24	79.9	0.05	0.5	0.025	0.25
	25	81.1	1.25	1.5	1.875	2.25
	26	81.2	1.35	2.5	3.375	6.25
	27	81.8	1.95	3.5	6.825	12.25
	28	82.8	2.95	4.5	13.275	20.25
	29	83.5	3.65	5.5	20.075	30.25
n=12						
sum	282	958.2			90.8	143
mean	23.5	79.85				
		b1=	0.634965		b0=	64.92832168



3. It is natural to expect that the larger the house, the higher the price. That is, we expect price and size of the house to be positively correlated. Ten houses were randomly selected among A local Danish newspaper ad for houses. The relationship between area and price is vaguely *suggesting* that it is not only the size of a house that determines the price. Your task is to investigate if there is any evidence to this suggestion.

size (m^2)	price 1000s DKK
104	1128
96	926
121	1284
145	701
62	712
96	851
149	976
196	1214
121	888
125	1272

	size (m^2) (x)	price 1000s DKK (y)	y-mean y	x-mean x	(x-mean x)(y-mean y)	(x-mean x)^2
	104	1128	132.8	-17.5	-2324	306.25
	96	926	-69.2	-25.5	1764.6	650.25
	121	1284	288.8	-0.5	-144.4	0.25
	145	701	-294.2	23.5	-6913.7	552.25
	62	712	-283.2	-59.5	16850.4	3540.25
	96	851	-144.2	-25.5	3677.1	650.25
	149	976	-19.2	27.5	-528	756.25
	196	1214	218.8	74.5	16300.6	5550.25
	121	888	-107.2	-0.5	53.6	0.25
	125	1272	276.8	3.5	968.8	12.25
n=10						
sum	1215	9952	-4.54747E-13	0	29705	12018.5
mean	121.5	995.2				
		b1=	2.471606274		b0=	694.8998378



Data appears very poorly correlated with no apparent linear relationship. It is not clear if the data has taken into account locality!

1. The table below gives a description of some sorting algorithms.

Name of sort	worst case	memory
bubble	$O(n^2)$	$O(1)$
selection	$O(n^2)$	$O(1)$
insertion	$O(n^2)$	$O(1)$
shell	$O(n \log^2 n)$	$O(1)$
binary tree	$O(n \log n)$	$O(n)$
merge	$O(n \log n)$	$O(n)$
heap	$O(n \log n)$	$O(1)$
quick	$O(n^2)$	$O(n \log n)$

a) for a list of n elements that need to be sorted, comment of the efficiency of the algorithms. Which is the best, worst and which would you recommend in that case (give reasons).
   i) n = 10
   ii) n = 10000000

Name of sort	worst case	n=10	n=10000000
bubble	$O(n^2)$	100	$1x10^{14}$
selection	$O(n^2)$	100	$1x10^{14}$
insertion	$O(n^2)$	100	$1x10^{14}$
shell	$O(n \log^2 n)$	10	490000000
binary tree	$O(n \log n)$	10	70000000
merge	$O(n \log n)$	10	70000000
heap	$O(n \log n)$	10	70000000
quick	$O(n^2)$	100	$1x10^{14}$

i) n = 10 shell or heap as computation cost and memory cost is lowest
ii) n = 10000000 heap as computation cost and memory cost is lowest

4. A square matrix has the same number of rows and columns and its size is defined by the variable **n**. The code fragment below performs the multiplication of two square matrices **A** and **B** and stores the result in matrix **C**.

```
int i,j,k,n;
// process rows
for (i = 0, i < n; i++)
  // process columns
  for (j = 0; j < n; j++)
    {
    c[i][j] = 0.0;
    // process row-column interactions and sum them into array c
    for (k = 0; k < n; k++)
      {
      c[i][j] = c[i][j] + a[i][k] * b[k][j];
      }
```

a) Using the code given, step through the statements (as though you were debugging the code) and compute the matrix **C** given that

$$A = \begin{pmatrix} 3 & 5 & 2 \\ 4 & 6 & 1 \\ 1 & -3 & 1 \end{pmatrix} \quad B = \begin{pmatrix} 2 & 1 & 1 \\ 5 & 6 & -3 \\ 3 & 4 & 1 \end{pmatrix} \quad C = \begin{pmatrix} 37 & 41 & -10 \\ 41 & 44 & -13 \\ -10 & -13 & 11 \end{pmatrix}$$

b) What is the computational count ***in terms of n*** for the code. Do this for the case when

i) you do **not** include the cost of the loop process

```
0       for (i = 0, i < n; i++)
              // process columns
0             for (j = 0; j < n; j++)
                {
n^2             c[i][j] = 0.0;
                // process row-column interactions and sum them into array c
0               for (k = 0; k < n; k++)
                  {
3n^3              c[i][j] = c[i][j] + a[i][k] * b[k][j];
                  }
= 3n^3 + n^2
or ass*(n^3+n^2) + add*n^3 + mul*n^3
```

ii) you do include the cost of the loop process

```
1+2n    for (i = 0, i < n; i++)
              // process columns
n(1+2n)       for (j = 0; j < n; j++)
                {
n^2             c[i][j] = 0.0;
                // process row-column interactions and sum them into array c
n^2(1+2n)       for (k = 0; k < n; k++)
                  {
3n^3              c[i][j] = c[i][j] + a[i][k] * b[k][j];
                  }
= 1+2n + n(1+2n)+ n^2 + n^2(1+2n) + 3n^3 = 5n^3 + 4n^2 + 3n +1
or ass*(1 + n + 2n^2 + n^3)+ com*(n + n^2 + n^3)+ add*(n + n^2 + 2n^3)+ mul*(n^3)
```

c) Two CPUs CPU1 and CPU2 are being considered to process the matrix multiplication above. Below are the costs of performing standard computational operations:

operation	CPU1 cost (microseconds)	CPU2 cost (microseconds)
add	1	3
subtract	1	3
multiply	4	1
divide	4	1
assign	2	1
compare	1	2

Comparing both CPUs, what is the approximate time needed to compute the triple nested loop for a matrix of size

(i) 3

(ii) 1000

Which CPU is better suited to matrix multiplication?

	cpu1	cpu2
3	0.000311	0.000352
1000	9006.004002	10007.006

add=	1	3
sub=	1	3
mul=	4	1
div=	4	1
ass=	2	1
com=	1	2