Question A

1) Base case (n=0):

$$a_0 = 10 \cdot 5^0 = 10 \cdot 1 = 10$$

Thus, the claim holds for n=0.

Induction Hypothesis:

Assume the claim hold for n:

$$a_n = 10 \cdot 5^n$$

Induction Step:

We'll show that the claim holds for n+1:

$$a_{n+1} = 10 \cdot 5^{n+1}$$

By sequence definition,

$$a_{n+1} = 5 \cdot a_n$$

By the induction hypothesis, we get that

$$5 \cdot a_n = 5 \cdot 10 \cdot 5^n$$

By exponentiation rules, we get

$$10 \cdot 5^{n+1}$$

2) We'll prove by induction

Base Case (n=1):

$$\sum_{i=1}^{1} i \cdot 2^i = 1 \cdot 2^1 = 2 = (n-1) \cdot 2^{1+1} + 2$$

Thus, the claim holds for n=1.

Induction Hypothesis:

Assume the claim hold for n≥1:

$$\sum_{i=1}^{n} i \cdot 2^i = (n-1) \cdot 2^{n+1} + 2$$

Induction Step:

We'll show that the claim holds for n+1:

$$\sum_{i=1}^{n+1} i \cdot 2^i = (n+1-1) \cdot 2^{n+1+1} + 2 = n \cdot 2^{n+2} + 2$$

$$\sum_{i=1}^{n+1} i \cdot 2^i = \sum_{i=1}^{n} (i \cdot 2^i) + (n+1) \cdot 2^{n+1}$$

By the induction hypothesis

$$\sum_{i=1}^{n} (i \cdot 2^i) + (n+1) \cdot 2^{n+1} = (n-1) \cdot 2^{n+1} + 2 + (n+1) \cdot 2^{n+1}$$

By distributivity we get

$$(n-1) \cdot 2^{n+1} + 2 + (n+1) \cdot 2^{n+1} = (n-1+n+1) \cdot 2^{n+1} + 2 = (2n) \cdot 2^{n+1} + 2$$

By exponentiation rule, we get

$$(2n) \cdot 2^{n+1} + 2 = n \cdot 2^{n+2} + 2$$

3) We'll prove by induction

<u>Base Case (n=1):</u>

$1 = 2^0$

Thus, the claim holds for n=1.

<u>Induction Hypothesis:</u>

Assume that for every integer n≥1, all integers in the range $[1,2^n - 1]$ can be written as the sum of distinct elements from $\{2^0, 2^1, \dots, 2^{n-1}\}$.

<u>Induction Step:</u>

We'll show that all integers in the range $[1,2^{n+1} - 1]$ can be written as the sum of distinct elements from $\{2^0, 2^1, \dots, 2^n\}$.

First, we'll divide the range into 3 sections $[1,2^n - 1], [2^n], [2^n + 1,2^{n+1} - 1]$.

From the induction hypothesis, we get that every number in the range of $[1,2^n - 1]$can be written as the sum of distinct elements from $\{2^0, 2^1, \dots, 2^{n-1}\}$.

$2^n \in \{2^0, 2^1, \dots, 2^{n-1}\}$, therefore, it can be written as itself.

Notice that by exponentiation rules $2^{n+1} - 1 = 2 \cdot 2^n - 1 = 2^n + 2^n - 1$.

Therefore $[2^n + 1,2^{n+1} - 1] = [2^n + 1, 2^n + 2^n - 1]$. That is, from the previous two sections of the range, we get that all the integers in that sub-range can be represented by distinct elements from $[1,2^n - 1] + 2^n$.

Question B

1) The proof is incorrect.

In the induction step, in the expression $n + 1 = n + n - (n - 1)$, they proceed from the assumption that there are two non-negative numbers $i, j \le n + 1$ such that $i + j = n + 1$.

Since the base case is 0, there is an option that $n + 1 = 1$, In that case, the only non-negative number that is smaller than 1 is 0. Therefore, in that case, there are no 2 non-negative numbers that are smaller than n+1 and their sum is n+1. Thus, the proof is not valid in this case, hence the proof is incorrect.

2) The proof is incorrect.

In the induction step, they proceed with the assumption that there are $k \ge 2$ common points between line 1 and line 2. When looking at the case when $n = 3$, p1 and p2 lie on one line and p2 and p3 on another line. Since line 1 and line 2 share only one point we can't determine that p3 is also on line 1. Therefore, the proof is invalid in this case, hence the proof is incorrect.

3) The proof is correct

Question C

Since x is initialized to be 1 and the for loop multiply It by 2 every round and then the while loop divide it by 2 every round until it reaches 1 again, at the end of the function $x = 1$.

At initialization s=2, and on every iteration of the for loop, it adds to it the iteration numbers time the new value of x. Thus, at the end of the function $s = 2 + \sum_{i=1}^{n} i \cdot 2^i$.

At the beginning of the function, r is set to be equal to n. Then, at the while loop, it increases by 1 on each iteration. Since the while loop is run till x is not bigger than 1, and on each loop the value of x is divided by 2, at the of the function $r = 2n$.

Question D

1)

$lists\_merge(\ell_1, \ell_2)$

$p_1 \leftarrow head[\ell_1]$
$p_2 \leftarrow head[\ell_2\,]$
$create\_list\ \ell_3$
$p_3 \leftarrow head[\ell_3]$
$\textbf{while}\ p_1 \neq null\ or\ p_2 \neq null\ \textbf{do}$
$\quad \textbf{if}\ data[p_1] = null\ \textbf{do}$
$\quad\quad \textbf{if}\ p_3 = -1\ \textbf{do}$
$\quad\quad\quad p_3 \leftarrow p_2$
$\quad\quad \textbf{else}$
$\quad\quad\quad next[p_3] \leftarrow p_2$
$\quad\quad\quad p_3 \leftarrow next[p_3]$
$\quad\quad \textbf{end else}$
$\quad\quad p_2 \leftarrow next[p_2]$
$\quad \textbf{else if}\ p_2 = null\ \textbf{do}$
$\quad\quad \textbf{if}\ p_3 = -1\ \textbf{do}$
$\quad\quad\quad p_3 \leftarrow p_1$
$\quad\quad \textbf{else}$
$\quad\quad\quad next[p_3] \leftarrow p_1$
$\quad\quad\quad p_3 \leftarrow next[p_3]$
$\quad\quad \textbf{end else}$
$\quad\quad p_1 \leftarrow next[p_1]$
$\quad \textbf{else if}\ data[p_2] > data[p_1]\textbf{do}$
$\quad\quad \textbf{if}\ p_3 = -1\ \textbf{do}$
$\quad\quad\quad p_3 \leftarrow p_1$
$\quad\quad \textbf{else}$
$\quad\quad\quad next[p_3] \leftarrow p_1$
$\quad\quad\quad p_3 \leftarrow next[p_3]$
$\quad\quad \textbf{end else}$
$\quad\quad p_1 \leftarrow next[p_1]$
$\quad \textbf{else if}\ data[p_1] > data[p_2]\ \textbf{do}$
$\quad\quad \textbf{if}\ p_3 = -1\ \textbf{do}$
$\quad\quad\quad p_3 \leftarrow p_2$
$\quad\quad \textbf{else}$
$\quad\quad\quad next[p_3] \leftarrow p_2$
$\quad\quad\quad p_3 \leftarrow next[p_3]$
$\quad\quad \textbf{end else}$
$\quad\quad p_2 \leftarrow next[p_2]$
$\quad \textbf{else if}\ data[p_1] = data[p_2]\ \textbf{do}$
$\quad\quad \textbf{if}\ p_3 = -1\ \textbf{do}$
$\quad\quad\quad p_3 \leftarrow p_2$
$\quad\quad \textbf{else}$
$\quad\quad\quad next[p_3] \leftarrow p_2$
$\quad\quad\quad p_3 \leftarrow next[p_3]$
$\quad\quad \textbf{end else}$
$\quad\quad p_1 \leftarrow next[p_1]$
$\quad\quad p_2 \leftarrow next[p_2]$
$\quad \textbf{end if}$
$\textbf{end while}$
$\textbf{return}\ \ell_3$

2)

$sort\ list(\ell)$ % Gets a list that it first element is not the minimum element and all the other are sorted in in increasing order, and return sorted list

$p_1 \leftarrow head[\ell]$
$p_2 \leftarrow next[p_1]$
$create\_list\ new$
$create\_list\ temp_1$
$create\_list\ temp_2$
$head[\ell] \leftarrow p_1$
$p_3 \leftarrow head[temp_2]$
$\boldsymbol{while}\ p_2 \neq null\ \boldsymbol{do}$
  $next[p_3] \leftarrow p_2$
  $p_3 \leftarrow next[p_3]$
  $p_2 \leftarrow next[p_2]$
$\boldsymbol{end\ while}$
$new \leftarrow list\_marge(temp_1\ , temp_2)$
$\boldsymbol{return}\ new$

Question E

1) The Idea of each of the array dimensions are to describe the value of each node in the linked list and the next and previous nodes of each.
i.e., the first dimension will be the previous node index at the array, the second dimension will be the value of the node, and the third dimension will be the next node index at the array.

2)

$\ell.free$: $index\ of\ the\ first\ element\ in\ the\ free\ list$

$insertLast(\ell, k)$ % Insert key k to the tail of the list
$\boldsymbol{if}\ free[\ell] = -1\ \boldsymbol{do}$
  $n \leftarrow l.length$
  $new\ell \leftarrow arry\ of\ size[n \cdot 2][3]$
  $\boldsymbol{for}\ i \leftarrow 1\ to\ n\ \boldsymbol{do}$
    $\boldsymbol{for}\ j \leftarrow 1\ to\ 3\ \boldsymbol{do}$
      $new\ell[i][j] \leftarrow \ell[i][j]$
    $\boldsymbol{end\ for}$
  $\boldsymbol{end\ for}$
$\boldsymbol{end\ if}$
  $cell \leftarrow \ell.free$
  $\ell[cell][data] \leftarrow k$
  $\ell.free \leftarrow next[\ell.free]$
  $\ell[cell][next] \leftarrow \ell.free$
$\boldsymbol{end}$

3)

$\ell.free$: $index\ of\ the\ first\ element\ in\ the\ free\ list$
$\ell.head$: $index\ of\ the\ first\ element\ in\ the\ data\ list$

$deletePrevKey(\ell, k)$ % Delete the previous key of k from the list

$q \leftarrow \ell.head$
$\boldsymbol{while}\ \ell[q][data] \neq k\ \boldsymbol{do}$
$\quad q \leftarrow \ell[q][next]$
$\boldsymbol{end\ while}$
$\boldsymbol{if}\ \ell[q][previous] \neq -1\ \boldsymbol{do}$
$\quad prev \leftarrow \ell\big[\ell[q][previous]\big][previous]$ % Get the index of the previous of the previous of k
$\quad \ell[prev][next] \leftarrow q$
$\boldsymbol{end\ if}$