Question 25

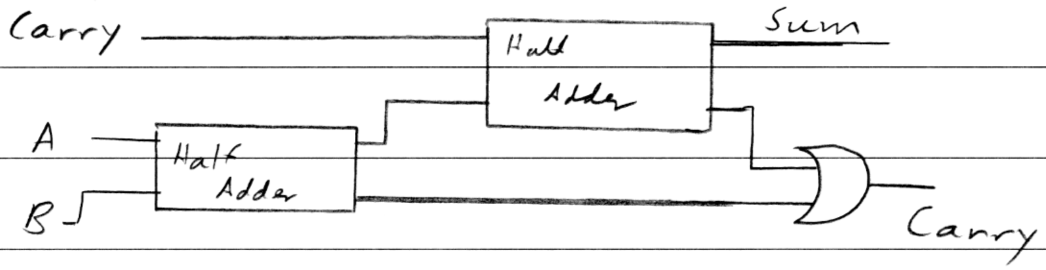
BOARD OF STUDIES
NEW SOUTH WALES

25

(a) (i)

A	B	C	S
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

(ii)



If two half ~~add~~ adders are connected like above, with ~~one~~ the sum of the first ~~a~~ half-adder feeding into the input of the second and the carry of both feeding into an ~~or~~ OR gate, a full ~~add~~ adder would have been created.

(b) Integer number representation occupies ~~one~~ a set memory amount and only representats whole positive or negative numbers. The integer data type cannot store fractions or parts of a whole. This would be suitable for simple mathematical processes where accuracy is not important, ~~or~~ or in control structures such as counters for loops.

Floating point on the other hand occupies more memory than integer does, it takes up either 32 or 64 bit of memory to handle (depending on precision). Though, its ability to store ~~decimal~~ fractions or parts of numbers lends ~~it~~ this data type to be used in scientific processess or mathematical process where accuracy is needed.

(c)

(i) 50 millimetres to the right,

80 millimetre ~~to the~~ up

(ii) ~~101100101~~

~~110100101~~

```
  ¹1¹0¹110¹010 +
   11010011
  ----------
  110000101
```

Add = 110000101

13_{10} = 1101

```
1101 = 0010 +
          1
       ----
       0011

       1100 +
       0011
       ----
       1111
```

~~1101) 110000101~~

```
        1
      ___________________________________
1101) 256 128 64  32  16  8  4  2  1
       1   1   0   0   0  0  1  0  1
       1   1   0   1
      ---------------
       1   0   1   1
```

Remainder = ~~1100~~

= ~~00001100~~

= 1100

```
(iii) BEGIN Extract (packet)
        CASEWHERE packet
            = 1 : packetLn = mid (StringIn, 2, 8)
            = 2 : packetLn = mid (StringIn, 12, 8)
            = 3 : packetLn = mid (StringIn, 26, 4)
        END CASEWHERE
      END

BEGIN CheckString
    IF Length (StringLn) = 30
        THEN stringvalid = 'true'
    ELSE
        ~~Stra~~ stringvalid = 'false'
    ENDIF
END

BEGIN Move (x, y)
    IF mid (x, 1, 1) = '0' THEN
        X direction = 'left'
    ELSE
        X direction = 'right'
    ENDIF
    IF mid (Y, 1, 1) = 'O' THEN
       Y direction = 'down'
    ELSE
       Y direction = 'Up'
    ENDIF
    Move Car X direction, mid (X, 2, 7) mm
    Move Car Y direction, mid (Y, 2, 7) mm
END

~~BEGIN MAIN PROGRAM~~
    ~~Check String~~

BEGIN CheckSum
    ~~result = mid (E~~
    ~~result = mid (X,~~
    result = (X + Y) / 13
    IF result = chk
       THEN string valid = 'true'
    ELSE
       string valid = 'false'
END
```

```
BEGIN MAIN PROGRAM
      x = Extract (1)
      y = Extract (2)
    chk = Extract (3)
    checksum
    checkstring
    IF stringvalid = 'true'
      THEN Move (x, y)
    END IF
END MAIN PROGRAM.
```