**DAILY ASSESSMENT FORMAT**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date:** | **13/July/2020** | **Name:** | **Divyashri Bahubali Samajage** |
| **Course:** | **Ames** | **USN:** | **4al17ec031** |
| **Topic:** | **Ames revision** | **Semester & Section:** | **6th A** |
| **GitHub Repository:** | **Divyashri\_course** |  |  |

|  |
| --- |
| **FORENOON SESSION DETAILS** |
| **Image of session** |
| **Report – Report can be typed or hand written for up to two pages.** The Thumb-2 Technology and Instruction Set Architecture: The Thumb-23 technology extended the Thumb Instruction Set Architecture (ISA) into a highly efficient and powerful instruction set that delivers significant benefits in terms of ease of use, code size, and performance (see Figure 1). The extended instruction set in Thumb-2.   Architecture of ARM Cortex M3: The Cortex™-M3 is a 32-bit microprocessor. It has a 32-bit data path, a 32-bit register bank, and 32-bit memory interfaces (see Figure 2). The processor has a Harvard architecture, which means that it has a separate instruction bus and data bus. This allows instructions and data accesses to take place at the same time, and as a result of this, the performance of the processor increases because data accesses do not affect the instruction pipeline.   Special Registers: The Cortex-M3 processor also has a number of special registers. They are as follows:   * + Program Status registers (PSRs)   + Interrupt Mask registers (PRIMASK, FAULTMASK, and BASEPRI)   + Control register (CONTROL)   Special registers can only be accessed via MSR and MRS instructions; they do not have memory addresses: |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Date:** | **13/July/2020** | **Name:** | **Divyashri Bahubali Samajage** | |
| **Course:** | **Mathematics for Machine**  **Learning: Linear Algebra** | **USN:** | **4al17ec031** | |
| **Topic:** | **Week 1** | **Semester Section:** | **6th A** | |
| **Git hub repository** | **Divyashri\_course** |  |  | |
| **AFTERNOON SESSION DETAILS** | | | |
| **Image of session** | | | |
| **Report – Report can be typed or hand written for up to two pages.**  consider a vector v whose initial point is the ***origin*** in an xy - coordinate system and whose terminal point is . We say that the vector is in ***standard position*** and refer to it as a position vector. Note that the ordered pair defines the vector uniquely. Thus we can use to denote the vector. To emphasize that we are thinking of a vector and to avoid the confusion of notation with ordered - pair and interval notation, we generally write v = < a, b >.  The coordinate a is the *scalar****horizontal component*** of the vector, and the coordinate b is the *scalar****vertical component*** of the vector. By **scalar**, we mean a *numerical* quantity rather than a *vector* quantity. Thus, is considered to be the **component form** of v. Note that a and b are NOT vectors and should not be confused with the vector component definition.  Now consider  with A = (x1, y1) and C = (x2, y2). Let’s see how to find the position vector equivalent to . As you can see in the figure below, the initial point A is relocated to the origin (0, 0). The coordinates of P are found by subtracting the coordinates of A from the coordinates of C. Thus, P = (x2 - x1, y2 - y1) and the position vector is .  It can be shown that  and  have the same magnitude and direction and are therefore equivalent. Thus,  =  = < x2 - x1, y2 - y1 >.  The ***component form*** of  with A = (x1, y1) and C = (x2, y2) is  = < x2 - x1, y2 - y1 >.  **Example 1** Find the component form of  if C = (- 4, - 3) and F = (1, 5).  **Solution** We have  = < 1 - (- 4), 5 - (- 3) > = < 5, 8 >.  Note that vector  is equivalent to position vector  with as shown in the figure above.  Now that we know how to write vectors in component form, let’s restate some definitions. The length of a vector v is easy to determine when the components of the vector are known. For v = < v1, v2 >, we have |v|2 = v21 + v22          **Using the Pythagorean theorem** |v| = √v21 + v22.  The ***length***, or ***magnitude***, of a vector v = < v1, v2 > is given by |v| = √v21 + v22.  Two vectors are *equivalent* if they have the same magnitude and the same direction.  Let u = < u1, u2 > and v = < v1, v2 >. Then < u1, u2 > = < v1, v2 >          if and only if u1 = v1 and u2 = v2. Operations on Vectors To multiply a vector v by a positive real number, we multiply its length by the number. Its direction stays the same. When a vector v is multiplied by 2 for instance, its length is doubled and its direction is not changed. When a vector is multiplied by 1.6, its length is increased by 60% and its direction stays the same. To multiply a vector v by a negative real number, we multiply its length by the number and reverse its direction. When a vector is multiplied by 2, its length is doubled and its direction is reversed. Since real numbers work like scaling factors in vector multiplication, we call them ***scalars*** and the products kv are called ***scalar multiples*** of v.  For a real number k and a vector v = < v1, v2 >, the ***scalar product*** of k and v is kv = k.< v1, v2 > = < kv1, kv2 >. The vector kv is a ***scalar multiple*** of the vector v.  **Example 2** Let u = < - 5, 4 > and w = < 1, - 1 >.Find - 7w, 3u, and - 1w.  **Solution** - 7w = - 7.< 1, - 1 > = < - 7, 7 >, 3u = 3.< - 5, 4 > = < -15, 12 >, - 1w = - 1.< 1, - 1 > = < - 1, 1 >.  Now we can add two vectors using components. To add two vectors given in component form, we add the corresponding components. Let u = < u1, u2 > and v = < v1, v2 >. Then u + v = < u = < u1 + v1, u2 + v2 > | | | |