1) Banker's Algorithm in C along with a requirement checker.

Code:

```
#include <stdio.h>
int main()
{
   // P0, P1, P2, P3, P4 are the Process names here

   int n = 5,  ch, m = 3, i, j, k, avail[3], ind = 0, y = 0,  flag;
   // printf("So like, can you enter the number of processes. That'd be like so cool:\n");
   printf("Enter the number of processes:\n");
   scanf("%d", &n);                // Number of resources
   int f[n], alloc[n][3], max[n][3], ans[n], need[n][m];

   for(i = 0; i<n; i++)
   {
      printf("Please enter the A, B and C allocated values of the Process P%d:\n", i);
      scanf("%d %d %d", &alloc[i][0], &alloc[i][1], &alloc[i][2]);
   }
   printf("**************************************************\n");
   for(i = 0; i<n; i++)
   {
      printf("Please enter the A, B and C Max values of the Process P%d:\n", i);
      scanf("%d %d %d", &max[i][0], &max[i][1], &max[i][2]);
   }
   printf("**************************************************\n");
   printf("Please Enter the available Resources in A, B and C:\n");
   scanf("%d %d %d", &avail[0], &avail[1], &avail[2]);

   printf("Is there any extra requirement from any of the Processes? -1 is no:\n");
   scanf("%d", &ch);
```

```
if(ch == -1)
{
   for(i = 0; i<n; i++){

      if(i == ch)
      {
         for(j = 0; j<m; j++)
         {
            printf("Enter %d Value of P%d", j+1, i);
            scanf("%d", &ch);
            alloc[i][j] += ch;
         }
      }
   }
}

for (k = 0; k < n; k++)
   f[k] = 0;

for (i = 0; i < n; i++)
{
   for (j = 0; j < m; j++)
      need[i][j] = max[i][j] - alloc[i][j];
}

for (k = 0; k < 5; k++)
{
   for (i = 0; i < n; i++)
   {
      if (f[i] == 0)
      {
```

```
            flag = 0;
            for (j = 0; j < m; j++)
            {
               if (need[i][j] > avail[j])
               {
                  flag = 1;
                  break;
               }
            }
            if (flag == 0)
            {
               ans[ind++] = i;
               for (y = 0; y < m; y++)
                  avail[y] += alloc[i][y];
               f[i] = 1;
            }
         }
      }
   }
}
flag = 1;
for (int i = 0; i < n; i++)
{
   if (f[i] == 0)
   {
      flag = 0;
      printf("The following system is not safe");
      break;
   }
}
if (flag == 1)
{
```

```
        printf("Following is the SAFE Sequence\n");
        for (i = 0; i < n - 1; i++)
            printf(" P%d ->", ans[i]);
        printf(" P%d", ans[n - 1]);
    }
    return 0;
}
```

Output: