|  |  |  |  |
| --- | --- | --- | --- |
| ID (count) | File Name | Code Block | Equivalent Mutant |
| 1 | DecrementOperator.c | int a= 5, b, c;  b = a--;  c = a--;  printf("Value of b = %d\n Value of c = %d\n", b,c);  return 0; | Replace – with ++ |
| 2 | IncrementOperator.c | int a= 5, b, c;  b = a++ + ++a; //Right to left  c = ++a + a++; //Right to left  printf("Value of b = %d\n Value of c = %d\n", b,c);  return 0; | Changed order from:  b = a++ + ++a to  b = ++a + a++  and from  c = ++a + a++ to  c = a++ + ++a |
| 3 | FibonacciGeneration.c | int fib(int n)  {  // Base case defined  if (n <= 1){  return n;  }  // Recursive Calls to fib function  return fib(n - 1) + fib(n - 2);  } | Replace n <=1 with n<1 |
| 4 | DecimalToBinary.c | while (power > 0) {  if (num >= power) {  firstOne = 1;  printf("1");  num %= power;  } else if (firstOne == 1) {  printf("0");  }  power /= 2;  }  printf("\n");  } | Replace printf(“0”) with print(“1”) |
| 5 | sorted\_list.c | void gnome\_sort(int \*array, int size){  int i, tmp;  for(i=1; i<size; ){  if(array[i-1] <= array[i])  ++i;  else{  tmp = array[i];  array[i] = array[i-1];  array[i-1] = tmp;  --i;  if(i == 0)  i = 1; | Replace <= with >= |
| 6 | increment\_number.c | int main(void) {  int num = 0;  int increment = 1;.  while(num < 100) {  sleep(1); // Wait 1 second each time, num has it's Value incremented  num += increment; // Increment num  printf("%d\n", num); | Replace 0 with 1. |
| 7 | DigitalRoot.c | int main() {  unsigned int number, temp, droot = 0;  printf("Enter a positive number: ");  scanf("%u", &number);  temp = number;  while(temp != 0) {  int digit = temp % 10;  droot += digit;  temp /=10;  if(temp == 0 && droot > 9) {  temp = droot;  droot = 0;  } | Replace == with >= |
| 8 | DailyWageCalc.c | …  else if (hour >= 17 && hour <= 20)  amount = 340 + (hour - 16)\*60;  else if (hour >= 21 && hour <= 24)  amount = 580 + (hour - 20)\*80;  ------ | Include another else statement:  else  amount = 0; |
| 9 | BinarySearch.c | int binarySearch(int array[], int number, int start, int end) {  /\* if start index is get end index, check if that element is equals wanter nmber \*/  if(start >= end) {  return array[start] == number ? 0 : 1;  } | Replace >= with > |
| 10 | shaker.c | for(it = 0; it < 2; ++it) {  flag = 1;  for(i = start[it]; i != end[it]; i += inc[it])  if(a[i - 1] > a[i]) {  swap(a + i - 1, a + i);  flag = 0;  } | Replace > with >= |
| 11 | bogo.c | bool is\_sorted(int \*a, int n)  {  while ( --n >= 1 ) {  if ( a[n] < a[n-1] ) return false;  }  return true;  } | Replace < with > |
| 12 | bubble\_two.c | while (j > 0) {  for (i = 1; i < j; i++) {  if (x[i] > x[i - 1]) {  t = x[i];  x[i] = x[i - 1];  x[i - 1] = t; | Replace > with < |
| 13 | sum\_one.c | int sum (int a, int b) //function definition  {  int s;  s=a+b;  return s; | Change a+b to a+b+2. |
| 14 | countPositive.c | int countPositive (int x[], int length)  { /\*Effects: return the number of  positive elements in x.\*/  int count = 0;  int i;  for (i = 0; i < length; i++) {  if (x[i] >= 0) {  count++; | Replace >= with > |
| 15 | sum.c | nt sum(int x[], int size)  {  int s = 0;  int i;    for (i = 0; i < size; i++) {  s = s + x[i];  }  return s; | Replace 0 with 1 |
| 16 | oddOrPos.c | int oddOrPos (int x[], int length){  int count = 0;  int i;  for (i = 0; i < length; i++) {  if (x[i] % 2 == 1 || x[i] % 2 == -1 || x[i] > 0) {  count++;  }  } | Change if-statement from:  if (x[i] % 2 == 1 || x[i] % 2 == -1 || x[i] > 0)  to  if (x[i] % 2 == 1 || x[i] % 2 == -1) |
| 17 | max.c | int max(int x, int y)  {  if (x > y)  return x;  else  return y;  } | Change > to >= |
| 18 | max.c | int max(int x, int y)  {  if (x > y)  return x;  else  return y;  } | Change to:  if (y > x)  return y;  else  return x; |
| 19 | year.c | int main()  {  int i;  for (i=Jan; i<=Dec; i++)  printf("%d ", i);    return 0; | Change from <= to < |
| 20 | merge\_sort.c | void mergeSort(int a[], int beg, int end)  {  if (beg < end)  {  int mid = (beg + end) / 2;  mergeSort(a, beg, mid);  mergeSort(a, mid + 1, end);  merge(a, beg, mid, end);  } | Change mid + 1 to mid |
| 21 | merge\_sort.c | void printArray(int a[], int n)  {  int i;  for (i = 0; i < n; i++)  printf("%d ", a[i]);  printf("\n");  } | Change to i = n-1, i >= 0 |
| 22 | line\_count.c | fclose(file);  //8  totalLinesCount++;  //9  printf("Total number of lines are : %d\n", totalLinesCount);  return 0; | Change ++ to -- |
| 23 | RecursiveSelectionSort.c | for (i = low + 1; i <= high; i++) {  if (list[i] < min)  {  min = list[i];  indexOfMin = i;  }  } | Change < to > |
| 24 | CheckPalindrome.c | int isPalindrome(char s[]) {  /\* The index of the first character in the string \*/  int low = 0;  /\* The index of the last character in the string\*/  int high = strlen(s)-1;  while (low < high) {  if (s[low] != s[high]) | Change < to <= |
| 25 | findVal.c | int findVal(int numbers[], int length, int val)  {  int findVal = -1;  int i;    for (i = 0; i < length; i++)  if (numbers[i] == val)  findVal = i; | Change == to != |
| 26 | findVal.c | int findVal(int numbers[], int length, int val)  {  int findVal = -1;  int i;    for (i = 0; i < length; i++)  if (numbers[i] == val)  findVal = i; | Change -1 to 1 |
| 27 | DynamicTwoDArray  UsingArrayOfPointer.c | int main() {  int r = 3, c = 4, i, j, count;  int \* arr[r];  for (i = 0; i < r; i++)  arr[i] = (int \* ) malloc(c \* sizeof(int)); | Change < to <= |
| 28 | Diagonal-Difference.c | for (j=0;j<n;j++)  {  scanf("%d ", &arr[i][j]);  //Taking diagonal sum of the matrix arr from the both side  if(arr[i][j]>=-100 && arr[i][j]<=100)  {  if(i==j) | Change from 100 to 1000 |
| 29 | quicksort.c | int main(int argc, char \*argv[]){  int size,i;  size = argc - 1;  for(i=0;i<size;i++)  { | Change from < to <= |
| 30 | Heap.c | void sort(char \*pq[],int length)  {  int k;  for (k=length/2;k>=1;k--)  sink(pq,k,length);  while (length>1) | Change > to >= |