**Test Cases**

**R1**: Interfaccia utente

**R1.1 Inserimento di operandi e operatori**  
Test #1  
Input: “2+2i”;  
Output: visualizzazione nello stack dell’ operando “2 + 2i”;

Test #2  
Input: “44.04”;  
Output: visualizzazione nello stack dell’ operando “44.04”;

Test #3  
Input: “+”;   
Output: esecuzione dell’ operazione di addizione;

Test #4  
Input: “-”;   
Output: esecuzione dell’ operazione di sottrazione;

Test #5  
Input: “/”;   
Output: esecuzione dell’ operazione di divisione;

Test #6  
Input: “sqrt”;   
Output: esecuzione dell’ operazione di radice quadrata;

Test #7  
Input: “swap”;   
Output: esecuzione dell’ operazione di swap;

Test #8  
Input: “>a”;   
Output: esecuzione dell’ operazione di inserimento valore nella variabile “a”;

**R1.2 Visibilità elementi nello stack**

Test #1  
Input: “3i”;  
Output: stack contenente il valore “3i” visualizzabile a schermo;

Test #2  
Stack prima dell’ input: 3, 2i, 4, 0, 5, 32.2;  
Input: “2 - 3i”;  
Output: stack contenente il valore “3i” visualizzabile a schermo, insieme agli elementi precedenti, 3, 2i, 4, 0, 5, 32.2;

Test #3  
Stack prima dell’ input: 7, 8, 23, 10i, 11i, 32i, 3 + 3i, 14, 15.78, 44, 39i, 19, 19.1, 20.3, 21, i, 3i, 44i, 3, 2i, 4, 0, 5, 32.2;  
Input: “2 - 3i”;  
Output: stack che visualizza i valori 7, 8, 23, 10i, 11i, 32i, 3 + 3i, 14, 15.78, 44, 39i, 19, 19.1, 20.3, 21, i, 3i, 44i, 3, 2i, 4, 0, 5, 32.2, poiché è stata raggiunta la capacità massima;

**R1.4 Visibilità elementi da inserire**

Test #1  
Input: click del pulsante “2”;  
Output: aggiunta della cifra “2” al display, il quale mostra “2”;

Test #2  
Display prima dell’ input: “-”;  
Input: click del pulsante “5”;  
Output: aggiunta della cifra “5” al display, il quale mostra “-5”;

Test #3  
Display prima dell’ input: “3+”;  
Input: click del pulsante “i”;  
Output: aggiunta dell’ unità immaginaria “i” al display, il quale mostra “3+i”;

Test #4  
Input: click del pulsante “**√**”;  
Output: aggiunta dell’ operatore “sqrt” al display, il quale mostra “sqrt”;

Test #5  
Display prima dell’ input: “323.22”  
Input: click del pulsante “←”;  
Output: rimozione dell’ ultima cifra (“2”) al display, il quale mostra “323.2”;

**R1.5 Visibilità variabili**

Test #1  
Menù variabili prima dell’ input: “a”, “b”, “c”, “d”, ecc…, con variabili non allocate;  
Stack prima dell’ input: 2 + 2i, -i, 47.4i, 32i, -26;  
Input: click del pulsante “>x” con variabile selezionata “a” dal menù;  
Output: a = 2 + 2i;  
Stack dopo l’ output: 2 + 2i, -i, 47.4i, 32i, -26;

Test #2  
Menù variabili prima dell’ input: “a”, “b”, “c”, “d”, ecc…, con variabili non allocate;  
Stack prima dell’ input: 3, -3, 0, 86.4, 55.2i, 0.2;  
Input: click del pulsante “>x” con variabile selezionata “e” dal menù;  
Output: e = 3;  
Stack dopo l’ output: 3, -3, 0, 86.4, 55.2i, 0.2;

Test #3  
Menù variabili prima dell’ input: “a”, “b”, “c”, “d”, ecc…, con variabili non allocate;  
Stack prima dell’ input: -26i, 4, 2, -16, 36, i, -i, -2i, 0, 4, 4 + 4i, 5, 6 - i, 3, 2, -2, 0, 1, -i;  
Input: click del pulsante “>x” con variabile selezionata “k” dal menù;  
Output: k = -26i;  
Stack dopo l’ output: -26i, 4, 2, -16, 36, i, -i, -2i, 0, 4, 4 + 4i, 5, 6 - i, 3, 2, -2, 0, 1, -i;

**R2**: Operazioni matematiche di base: la calcolatrice deve poter svolgere le operazioni matematiche di base, quali addizione, sottrazione, moltiplicazione, divisione, radice quadrata e inversione di segno.

**R2.1 Operazione di addizione** (*per i seguenti test l’ ordine dello stack è da intendersi in ordine dal top, elemento in cima, all’ ultimo elemento*)  
Test #1 (*numero reale + numero immaginario*)  
Stack prima dell’ input: 11.3, 2i, 53i, -4i, 2i, 28.32i, 0, 2i, 53i, -4i, 2i, 2 + 2i, -4i, 2i, 2 + 2i;  
Input: “+”;  
Stack aggiornato in output: 11.3 + 2i, 53i, -4i, 2i, 28.32i, 0, 2i, 53i, -4i, 2i, 2 + 2i, -4i, 2i, 2 + 2i;

Test #2 (*numero complesso + numero immaginario*)  
Stack prima dell’ input: 11.3 + 2i, 53i, -4i, 2i, 28.32i, 0, 2i, 53i, -4i, 2i, 2 + 2i, -4i, 2i, 2 + 2i;  
Input: “+”;  
Stack aggiornato in output: 11.3 + 55i, -4i, 2i, 28.32i, 0, 2i, 53i, -4i, 2i, 2 + 2i, -4i, 2i, 2 + 2i;

Test #3 (*numero complesso + numero immaginario negativo*)  
Stack prima dell’ input: 11.3 + 55i, -4i, 2i, 28.32i, 0, 2i, 53i, -4i, 2i, 2 + 2i, -4i, 2i, 2 + 2i;  
Input: “+”;  
Stack aggiornato in output: 11.3 + 51i, 2i, 28.32i, 0, 2i, 53i, -4i, 2i, 2 + 2i, -4i, 2i, 2 + 2i;

Test #4 (*numero complesso negativo + numero reale negativo*)  
Stack prima dell’ input: -22.963 - i, -69.3, 522.083i;  
Input: “+”;  
Stack aggiornato in output: -92.263 – i, 522.083i;

Test #5 (*numero reale negativo + numero reale positivo*)  
Stack prima dell’ input: -543.076, 825,4;  
Input: “+”;

Stack in output: 282.324;

Test #6 (*numero immaginario negativo + numero immaginario negativo*)  
Stack prima dell’ input: -55.2i, -i, 23, 4 + 2.3i;  
Input: “+”;  
Stack aggiornato in output: -56.2i, 23, 4 + 2.3i;

Test #7 (*zero + numero reale positivo*)  
Stack prima dell’ input: 0, 22.421, 52.89;  
Input: “+”;  
Stack aggiornato in output: 22.421;

Test #8 (*zero + numero immaginario negativo*)  
Stack prima dell’ input: 0, -633i;  
Input: “+”;  
Stack aggiornato in output: -633i;

Test #9 (*numero complesso negativo + zero*)  
Stack prima dell’ input: 0, -52 – 9.28i, 432, 22.76i, 55.34;  
Input: “+”  
Stack aggiornato in output: -52 – 9.28i, 432, 22.76i, 55.34;

**R2.2 Operazione di sottrazione**

Test #1 (*numero reale - numero reale*)  
Stack prima dell’ input: 54.8, 36, i, 22, 12.3i;  
Input: “-”;  
Stack aggiornato in output: 18.8, i, 22, 12.3i;

Test #2 (*numero reale – numero immaginario*)  
Stack prima dell’ input: 32, 554.8i, 32 - 4i, 21.7;  
Input: “-”;  
Stack aggiornato in output: 32 - 554.8i, 32 - 4i, 21.7;

Test #3 (*numero complesso – numero complesso negativo*)  
Stack prima dell’ input: 5 + 3i, -65 - 87i;  
Input: “-”;  
Stack aggiornato in output: 70 + 90i;

Test #4 (*numero reale – zero*)  
Stack prima dell’ input: 25, 0, 34.3, -2i, 55.4i, 2 + 3i;  
Input: “-”;  
Stack aggiornato in output: 25, 34.3, -2i, 55.4i, 2 + 3i;

Test #5 (*zero – numero complesso*)  
Stack prima dell’ input: 0, 6 + 89i, 23.55, 4i, -2;  
Input: “-”;  
Stack aggiornato in output: -6 - 89i, 23.55, 4i, -2;

Test #6 (*numero immaginario – numero complesso*)  
Stack prima dell’ input: 74i, 6 - 8i, 0, 1.2, -23;  
Input: “-”;  
Stack aggiornato in output: -6 + 82i, 0, 1.2, -23;

**R2.3 Operazione di moltiplicazione**

Test #1 (*numero reale \* numero reale*)  
Stack prima dell’ input: 5, 8, i, 12.3, 3 + 4i;  
Input: “\*”;  
Stack aggiornato in output: 40, i, 12.3, 3 + 4i;

Test #2 (*numero reale \* numero immaginario*)  
Stack prima dell’ input: 12.02, 8.3i, 40, 2 - 2i;  
Input: “\*”;  
Stack aggiornato in output: 99.766i, 40, 2 - 2i;

Test #3 (*numero complesso \* numero complesso*)  
Stack prima dell’ input: 3 - 56i, 5 + 2i, 54.26, -i, 1;  
Input: “\*”;  
Stack aggiornato in output: 127 - 274i, 54.26, -i, 1;

Test #4 (*zero \* numero immaginario*)  
Stack prima dell’ input: 0, -56.8i, 23, 12i, -34, 2.4 + 5i;  
Input: “\*”;  
Stack aggiornato in output: 0, 23, 12i, -34, 2.4 + 5i;

Test #5 (*numero immaginario \* numero complesso negativo*)  
Stack prima dell’ input: i, -6 - 9.8i, 0, 1, 3.3;  
Input: “\*”;  
Stack aggiornato in output: 9.8 - 6i, 0, 1, 3.3;

Test #6 (*numero reale negativo \* numero reale negativo*)  
Stack prima dell’ input: -53.8, -69.8, 9.8 - 6i, 0, 33.27, 11.11i;  
Input: “\*”;  
Stack aggiornato in output: 3755.24, 9.8 - 6i, 0, 33.27, 11.11i;

**R2.4 Operazione di divisione**

Test #1 (*numero reale / numero reale negativo*)  
Stack prima dell’ input: 32, -8, 11, 26i, -i, 0, 43.3;  
Input: “/”;  
Stack aggiornato in output: -4, 11, 26i, -i, 0, 43.3;

Test #2 (*numero complesso / numero immaginario*)  
Stack prima dell’ input: 2 + 2i, 6i, 14, 26i, -3;  
Input: “/”;  
Stack aggiornato in output: 0.333 - 0.333i, 16, 26i, -3;

Test #3 (*numero complesso negativo / numero complesso*)  
Stack prima dell’ input: -5 -7.4i, 9 + 5.8i, 13, 2, -4, -1, 2 + i, 34 - 33i, 27.8i, -3, -2;  
Input: “/”;  
Stack aggiornato in output: -0.767 - 0.328i, 13, 2, -4, -1, 2 + i, 34 - 33i, 27.8i, -3, -2;

Test #4 (*numero complesso negativo / numero complesso negativo*)  
Stack prima dell’ input: -4 - 6i, -4 - 6i, 23, 12.3i, 0, 22;  
Input: “/”;  
Stack aggiornato in output: 1, 23, 12.3i, 0, 22;

Test #5 (*zero / numero immaginario*)  
Stack prima dell’ input: 0, -i, 32, 54.3, -i, i, 12, 74 + 26.4i;  
Input: “/”;  
Stack aggiornato in output: 0, 32, 54.3, -i, i, 12, 74 + 26.4i;

Test #6 (*numero complesso / zero*)  
Stack prima dell’ input: 85 - i, 0, 12.4, 22i, 11 + 12i, 23i, -4, 2;  
Input: “/”;  
Stack aggiornato in output: MathError;

**R2.5 Operazione di radice quadrata**

Test #1 (*radice di un quadrato perfetto*)  
Stack prima dell’ input: 16, i, 2, 3 + i;  
Input: “sqrt”;  
Stack aggiornato in output: -4, 4, i, 2, 3 + i;

Test #2 (*radice di un numero reale*)  
Stack prima dell’ input: 212.8, -i, 3 + 47i, 12.53, 0, 2;  
Input: “sqrt”;  
Stack aggiornato in output: -14.588, 14.588, -i, 3 + 47i, 12.53, 0, 2;

Test #3 (*radice di un numero reale negativo*)  
Stack prima dell’ input: 65, 11.3i, 28.32i, 0, 2i, 53i, -4i, 2i, 2 + 2i, -4i, 2i, 2 + 2i;  
Input: “sqrt”;  
Stack aggiornato in output: -8.062i, 8.062i, 11.3i, 28.32i, 0, 2i, 53i, -4i, 2i, 2 + 2i, -4i, 2i, 2 + 2i;

Test #4 (*radice di un numero immaginario*)  
Stack prima dell’ input: 42i, 22i, 11 + 11i, 23.56i, 12.3 - i;  
Input: “sqrt”;  
Stack aggiornato in output: -4.583 - 4.583i, 4.583 + 4.4583i, 22i, 11 + 11i, 23.56i, 12.3 - i;

Test #5 (*radice di un numero complesso*)  
Stack prima dell’ input: 22 + 11i, 43.43, 27, -3, 3, 0, 43.42;  
Input: “sqrt”;  
Stack aggiornato in output: -4.827 - 1.139i, 4.827 + 1.139i, 43.43, 27, -3, 3, 0, 43.42;

**R2.6 Inversione di segno**

Test #1 (*numero reale*)  
Stack prima dell’ input: 3;  
Input: “+-”;  
Stack aggiornato dopo l’ output: -3;

Test #2 (*numero immaginario*)  
Stack prima dell’ input: 56.03i, 32, 33i, 2 + 4i, 5 + 6i, 12 + 23i;  
Input: “+-”;  
Stack aggiornato dopo l’ output: -56.03i, 32, 33i, 2 + 4i, 5 + 6i, 12 + 23i;

Test #3 (*numero complesso*)  
Stack prima dell’ input: 3 - 2i, 2i, 4i, 16, -16, 0, 32i, 35.65;  
Input: “+-”;  
Stack aggiornato dopo l’ output: -3 + 2i, 2i, 4i, 16, -16, 0, 32i, 35.65;