UNIVERSITÀ DEGLI STUDI DI URBINO CARLO BO

Corso di Laurea in Informatica Applicata

"Programma per la verifica delle congetture di Beal, Cramer e Collatz"

Relazione del progetto d'esame per l'insegnamento di Programmazione Procedurale

Sessione Invernale 2022/23

Balducci Milena Matricola 321791

1 Specifica del Problema

La congettura di Beal asserisce che se $a^x + b^y = c^z$ dove $a,b,c,x,y,z \in \mathbb{N}$ con $a,b,c \geq 1$ e $x,y,z \geq 3$, allora a,b,c hanno un fattore primo in comune.
La congettura di Collatz asserisce che la funzione $f : \mathbb{N}_{>0} \longrightarrow \mathbb{N}_{>0}$ definita ponendo $f(n) = n/2$ se n è pari e $f(n) = 3 \cdot n + 1$ se n è dispari genera 1 dopo un numero finito di applicazioni ai numeri mano a mano ottenuti.
La congettura di Cramer asserisce che il valore assoluto della differenza tra due numeri primi consecutivi è minore del quadrato del logaritmo naturale del più piccolo dei due numeri.
Scrivere un programma ANSI C che chieda all'utente quale congettura intende considerare e poi la verifica acquisendo dalla tastiera a,b,c,x,y,z nel primo caso (se non vale $a^x + b^y = c^z$, il programma lo stampa sullo schermo e poi verifica comunque se a,b,c hanno un fattore primo in comune e ne stampa l'esito sullo schermo), $n > 0$ nel secondo caso (il programma stampa sullo schermo tutti i numeri generati), due numeri primi consecutivi ≥ 11 nel terzo caso (il programma stampa sullo schermo sia il valore assoluto della differenza tra i due numeri che il quadrato del logaritmo naturale del più piccolo dei due numeri).

2 Analisi del Problema

2.1 Dati di Ingresso del Problema

I dati di ingresso sono l'operazione scelta dall'utente e, a seconda della scelta effettuata:

- tre potenze con base intera $b \geq 1$ e esponente intero $e \geq 3$;
- un numero intero n> 0;
- due numeri primi p_n e $p_{n+1} \geq 11$.

2.2 Dati di uscita del Problema

I dati di uscita del problema sono l'esito della verifica della congettura in esame e, in base all'operazione scelta dall'utente:

- il fattore primo comune alle tre basi scelte;
- la successione $\{n, f(n)...f^k(n)\}$ generata dalla funzione $f : \mathbb{N}_{>0} \longrightarrow \mathbb{N}_{>0}$;
- il valore assoluto della differenza tra i due numeri primi inseriti, e il quadrato del logaritmo naturale del minore dei due.

2.3 Relazioni intercorrenti tra i dati del Problema

- Un numero $p \in \mathbb{N}$ si dice primo se $\nexists q \in \mathbb{N} \setminus \{0,1\} : x \neq p \wedge x \; divide \; p$, ovvero se p ha come unici divisori 1 e se stesso. Per stabilire se un numero è primo o meno, è sufficiente verificare l'esistenza di eventuali divisori di p compresi tra 2 e $\sqrt{p}$, in quanto supponendo di controllare sequenzialmente tutti i numeri a partire da 2, se $\sqrt{p}$ non è un fattore primo di p, allora il più piccolo fattore primo di p sarebbe $\sqrt{p}+1$, e il più piccolo numero ottenibile con questo fattore è $(sqrt\, p+1)^2$, che è maggiore di p.
  Per verificare che due numeri primi p_n e p_{n+1} siano consecutivi, occcorre verificare se $\forall n \in (p_n, p_{n+1}) \; \exists x \in \mathbb{N} \setminus \{0,1\} \; : \; x \neq n \; \wedge \; x \; divide \; n$.

- I fattori primi di un numero $n \in \mathbb{N}$ sono tutti i numeri primi che lo dividono esattamente (ovvero con resto nullo).
  Per individuare i fattori primi comuni a due numeri $n, m \in \mathbb{N}$ è sufficiente individuare tutti i numeri primi $x \in \mathbb{N} : x \; divide \; n \; \wedge \; x \; divide \; m$.

- L'esito della verifica corrisponde al valore di verità della proposizione logica associata alla congettura che si è voluta verificare:
  - La congettura di Beal afferma che $\forall (a,b,c \geq 1), (x,y,z \geq 3) \in \mathbb{N} (a^x + b^y = c^z \Longrightarrow \exists q \in \mathbb{N} : q \; divide \; a \; \wedge \; q \; divide \; b \; \wedge \; q \; divide \; c)$.
  - La congettura di Collatz afferma che, data la successione $\{n, f(n), f^2(n)...f^k(n)\}$, $\forall n \in \mathbb{N} \setminus 0 \; \exists k \in \mathbb{N} : f^k(n) = 1$, con:

$$f(x) = \begin{cases} x/2, & \text{se } x \text{ pari} \\ 3 \cdot x + 1, & \text{se } x \text{ dispari} \end{cases}$$

  - La congettura di Cramer afferma che $\forall p_n, p_{n+1} \in \mathbb{N} (|p_n - p_{n+1}| < (\ln(p_n))^2)$, dove p_n è l'n-esimo numero primo.

  L'esito della verifica è positivo se, per i dati inseriti in ingresso, il valore logico della proposizione che formalizza la congettura scelta equivale a *vero*.

3 Progettazione dell'Algoritmo

3.1 Scelte di progetto

- Per semplificare la validazione dei dati di ingresso nella congettura di Cramer, che richiede di verificare che i due numeri primi inseriti siano consecutivi, si è deciso di imporre dei limiti all'utente, acquisendo prima il numero primo minore e poi quello maggiore.
- Altri limiti che si è deciso di imporre all'utente riguardano l'acquisizione dei dati di input della congettura di Beal: dal momento che la verifica della congettura richiede di calcolare la potenza di due numeri, si è deciso di verificare che le potenze calcolate non superino il valore massimo del tipo di dato utilizzato, riacquisendo i dati in caso contrario.
- Per evitare ridondanze di codice, si è deciso inoltre di definire due funzioni per svolgere operazioni che si ripetono più volte all'interno dell'algoritmo.
  - La prima è la funzione per l'acquisizione e la validazione stretta dei dati di ingresso, che prende come parametri il valore da acquisire (passato per indirizzo), un messaggio specifico da stampare in base all'operazione scelta e il valore minimo che il dato da acquisire può assumere. Se il valore minimo è 11, e quandi si stanno acquisendo i dati in ingresso della congettura di Cramer, la funzione verifica anche che il valore inserito sia un numero primo.
  - La seconda funzione si occupa di verificare che un numero sia primo, sfruttando la relazione descritta nel paragrafo precedente, e prende come parametro il valore da acquisire, mentre restituisce 1 se il numero è primo, 0 se non lo è.
- Un'altra scelta effettuata per evitare ridondanze è quella di assegnare l'esito complessivo della verifica ad una variabile di tipo numerico dichiarata nella funzione main, in modo da poter comunicare l'esito complessivo della verifica all'interno di quest' ultima, evitando le ripetizioni di codice che si verrebbero a creare comunicando l'esito di ogni verifica all'interno delle singole funzioni.

3.2 Passi dell'algoritmo

I passi dell'algoritmo per risolvere il problema sono i seguenti:

- Acquisire l'operazione scelta dall'utente.
- In base alla scelta effettuata:
  - Acquisire le tre basi e i tre esponenti;
  - Verificare se $a^x + b^y = c^z$ è soddisfatta dai valori inseriti;
  - Verificare se le tre basi hanno fattori primi in comune;
  - Stampare a schermo l'eventuale fattore primo individuato.
- oppure:
  - Acquisire un numero intero maggiore di zero;
  - Verificare se il numero acquisito è pari o dispari;
  - Applicare la funzione al valore inserito;
  - Stampare il nuovo valore;
  - Ripetere l'applicazione ai valori mano a mano ottenuti, finchè diversi da 1;
- oppure:
  - Acquisire i due numeri primi consecutivi;

  - Calcolare il valore assoluto della differenza dei due numeri;
  - Calcolare il quadrato del logaritmo naturale del numero minore;
  - Verificare se quest'ultimo è maggiore del valore assoluto della differenza tra i due numeri;
  - Stampare i due valori ottenuti;
- Stampare l'esito della verifica.

4 Implementazione dell'algoritmo

4.1 Codice Sorgente

```
/*********************************************************************/
/* PROGRAMMA PER VERIFICARE LE CONGETTURE DI BEAL, COLLATZ E CRAMER */
/*********************************************************************/

/* INCLUSIONE DELLE LIBRERIE */
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include <limits.h>

/* DICHIARAZIONE DELLE FUNZIONI */
int acquisisci_operazione(void);
void acquisisci_valori(unsigned int *,
                       char *,
                       int);
int primo(unsigned int);
int verifica_beal(void);
int verifica_collatz(void);
int verifica_cramer(void);

/*****************************/
/* DEFINIZIONE DELLE FUNZONI */
/*****************************/

/* DEFINIZIONE DELLA FUNZIONE MAIN */
int main(void)
{
   /* Dichiarazione variabili locali alla funzione*/
   int op;          /* INPUT: operazione scelta */
   int esito;       /* OUTPUT: esito della verifica */

   /* Stampare il titolo del programma */
   printf("PROGRAMMA PER LA VERIFICA DI CONGETTURE\n");
   putchar('\n');

   /* Acquisire l'operazione scelta */
   op = acquisisci_operazione();

   /* Verificare la congettura */
   switch (op)
   {
     case 1:
       esito = verifica_beal();
       break;
     case 2:
       esito = verifica_collatz();
       break;
     case 3:
       esito = verifica_cramer();
       break;
     default:
       esito = 0;
```

```
        break;
    }

    /* Stampare l'esito complessivo della verifica */
    putchar('\n');
    printf("Congettura %s.\n",
           (esito)?
           "verificata":
           "non verificata");

    return(0);
}

/* DEFINIZIONE DELLA FUNZIONE PER SCEGLIERE LA CONGETTURA DA VERIFICARE */
int acquisisci_operazione(void)
{
    /* Dichiarazione delle variabili locali alla funzione */
    int operazione;          /* INPUT/OUTPUT: operazione scelta */
    int esito_acquisizione,  /* LAVORO: esito della scanf */
        acquisizione_errata; /* LAVORO: esito globale dell'acquisizione */

    do
    {
      /* Stampare le istruzioni per la selezione */
      putchar('\n');
      printf("Scegliere una congettura da verificare:\n\n");
      printf("(1) - Beal\n");
      printf("(2) - Collatz\n");
      printf("(3) - Cramer.\n");
      putchar('\n');
      printf("Digitare la propria scelta e premere invio: ");

      /* Acquisire l'operazione scelta */
      esito_acquisizione = scanf("%d",
                                 &operazione);

      acquisizione_errata = esito_acquisizione != 1 || operazione < 1 || operazione > 3;

      /* Stampare un messaggio di errore */
      if (acquisizione_errata)
        printf("Valore non accettabile!\n");
      while (getchar() != '\n');
    }
    while (acquisizione_errata);

    return(operazione);
}

/* DEFINIZIONE DELLA FUNZIONE PER VERIFICARE I NUMERI PRIMI */
int primo(unsigned int valore) /* INPUT: il valore da verificare */
{
    /* Dichiarazione delle variabili locali */
    int primo;      /* OUTPUT: esito della verifica */
    unsigned int i; /* LAVORO: indice di scorrimento */

    /* Verificare se il valore ha divisori */
    for (i = 2,
```

```
          primo = 1;
        i < sqrt(valore) && primo != 0;
        i++)
  {
      if(valore % i == 0)
         primo = 0;
  }

   return(primo);
}

/* DEFINIZIONE DELLA FUNZIONE PER ACQUISIRE VALORI NUMERICI */
void acquisisci_valori(unsigned int *valore, /*INPUT/OUTPUT: valore da acquisire */
                       char *messaggio,      /* LAVORO: messaggio specifico */
                       int min)              /* LAVORO: valore minimo */
{
   /* Dichiarazione delle variabili locali */
   int esito_acquisizione,  /* LAVORO: esito della scanf*/
       acquisizione_errata; /* LAVORO: esito complessivo dell' acquisizione */

   do
   {
     /* Stampare il messaggio specifico */
     putchar('\n');
     printf("Digita %s:  ",
            messaggio);

     /* Acquisire il valore */
     esito_acquisizione = scanf("%u",
                                valore);

     /* Verificare l'esito dell'acquisizione */
     if (min >= 11)
       acquisizione_errata = esito_acquisizione != 1 || *valore < min || primo(*valore) != 1;
     else
       acquisizione_errata = esito_acquisizione != 1 || *valore < min;

     /* Stampare un messaggio di errore */
     if (acquisizione_errata)
       printf("Valore non accettabile!\n");
     while (getchar() != '\n');
   }
   while (acquisizione_errata);
}

/* DEFINIZIONE DELLA FUNZIONE PER VERIFICARE LA CONGETTURA DI BEAL */
int verifica_beal(void)
{
   /* Dichiarazione delle variabili locali */
   unsigned int basi[3], /* INPUT: basi delle tre potenze */
                esp[3]; /* INPUT: esponenti delle tre potenze */
   int i;     /* LAVORO: indice di scorrimento */
   int div;      /* LAVORO: divisore per la scomposizione in fattori */
   int esito;    /* OUTPUT: esito complessivo della verifica */
   int fattore_comune; /*OUTPUT: fattore comune trovato */

   /* Stampare la scelta effettuata */
```

```
printf("\nCONGETTURA DI BEAL.\n");

/* Acquisire le tre potenze */
for (i = 0;
     i < 3;
     i++)
{
  do{
  acquisisci_valori(basi + i,
                    "una base (>= 1)",
                    1);

  acquisisci_valori(esp + i,
                    "un esponente (>= 3)",
                    3);

   if (pow(basi[i], esp[i]) >= UINT_MAX)
   {
     printf("La potenza inserita e' troppo elevata!\n");
     printf("Prego reinserirla.\n");
   }
  }
  while(pow(basi[i], esp[i]) >= UINT_MAX);
}

/* Cercare eventuali fattori primi comuni */
for (fattore_comune = 0,
       div = 2;
     fattore_comune == 0;
     div++)
 {
   if (primo(div) == 1)
   {
     if (basi[0] % div == 0 && basi[1] % div == 0 && basi[2] % div == 0)
       fattore_comune = div;
   }
 }

 if ((pow(basi[0], esp[0]) + pow(basi[1], esp[1])) == pow(basi[2], esp[2]))
 {
    printf("\nL'equazione a^x + b^y = c^z e' soddisfatta dai valori inseriti.\n");

    if (fattore_comune != 0)
   {
     printf("E' stato trovato un fattore primo comune: %d\n",
            fattore_comune);
     esito = 1;
   }
   else
   {
     printf("Non sono stati trovati fattori primi comuni.\n");
     esito = 0;
   }
 }
 else
 {
   printf("\nL'equazione a^x + b^y = c^z non e' soddisfatta dai valori inseriti.\n");
```

```
      esito = 1;

      if (fattore_comune != 0)
      {
        printf("E' comunque stato trovato un fattore primo comune: %d\n",
               fattore_comune);
      }
      else
      {
        printf("Non sono stati trovati fattori primi comuni.\n");
      }
    }

   return(esito);
}

/* DEFINIZIONE DELLA FUNZIONE PER VERIFICARE LA CONGETTURA DI COLLATZ */
int verifica_collatz(void)
{
   /* Dichiarazione delle variabili locali alla funzione */
   unsigned int val_corr; /* INPUT: valore inserito dall'utente */
   int verifica;          /* OUTPUT: esito della verifica */

   /* Stampare la scelta effettuata */
   printf("\nCONGETTURA DI COLLATZ.\n");

   /* Acquisire il valore di partenza */
   putchar('\n');
   acquisisci_valori(&val_corr,
                     "un numero intero (> 0)",
                     1);

   do
   {
      /* Verificare se il valore e' pari */
      if (val_corr % 2 == 0)
      {
        val_corr = val_corr / 2;
      }
      else
      {
        val_corr = 3 * val_corr + 1;
      }

      printf("%d ",
             val_corr);
   }
   while (val_corr != 1);

   /* Stabilire l'esito della verifica */
   verifica = (val_corr == 1)?
              1:
              0;

   return(verifica);
}
```

```
/* DEFINIZIONE DELLA FUNZIONE PER VERIFICARE LA CONGETTURA DI CRAMER */
int verifica_cramer(void)
{
   /* Dichiarazione delle variabili locali */
   unsigned int primo1, /* INPUT: primo numero primo*/
                primo2; /* INPUT: secondo numero primo*/
   int i,         /* LAVORO: indice di scorrimento */
       acquisizione_errata;  /* LAVORO: esito acquisizione */
   unsigned int abs_diff;  /* OUTPUT: valore assoluto della differenza */
   double  pow_ln;    /* OUTPUT: quadrato del ln del numero minore */
   int verifica;/* OUTPUT: esito della verifica */

   /* Stampare la scelta effettuata */
   printf("\nCONGETTURA DI CRAMER.\n");
   putchar('\n');

   do
   { /* Acquisire il primo numero primo */
     acquisisci_valori(&primo1,
                       "il primo numero primo (>= 11)",
                       11);

     /* Acquisire il secondo numero primo */
     acquisisci_valori(&primo2,
                       "il secondo numero primo (maggiore del precedente)",
                       primo1);


     /* Verificare che siano consecutivi */
     for(i = (primo1 + 1),
           acquisizione_errata = 0;
         i < primo2;
         i++)
     {
       if (primo(i) == 1)
         acquisizione_errata = 1;

     }

     /* Stampare messaggio di errore */
     if (acquisizione_errata)
       printf("\nI due numeri devono essere consecutivi!\n");
   }
   while (acquisizione_errata);

   /* Calcolare il valore assoluto della differenza */
   abs_diff = abs(primo1 -
                  primo2);

   /* Calcolare il quadrato del logaritmo naturale */
   pow_ln = pow(log(primo1),
                2);

   /* Stabilire l'esito della verifica */
   verifica = abs_diff < pow_ln;

   /* Stampare i risultati */
```

```
   printf("Il valore assoluto della differenza tra %u e %u e' %u.\n",
          primo1,
          primo2,
          abs_diff);

   printf("Il quadrato del logaritmo naturale del numero minore vale %.2f.\n",
          pow_ln);

   printf("Il valore assoluto della differenza %s minore del quadrato del logaritmo di %d.\n",
          (verifica)?
          "e'":
          "non e'",
          primo1);

   return(verifica);
}
```

4.2 Makefile

```
#Makefile del programma per verificare le congetture di Beal, Collatz e Cramer.

verifica_congetture: verifica_congetture.c Makefile
  gcc -ansi -Wall -O verifica_congetture.c -o verifica_congetture -lm

pulisci:
  rm -f verifica_congetture.o

pulisci_tutto:
  rm -f verifica_congetture.o verifica_congetture
```

5 Testing del programma

I test effettuati rivelano che se l'utente sceglie una operazione diversa da 1, 2 o 3, il programma emette un messaggio di errore e chiede di rieffettuare la scelta.
Allo stesso modo, il programma acquisisce correttamente i valori in ingresso in base alle limitazioni imposte dall'operazione scelta dall'utente; e rieffettua l'acquisizione degli input della congettura di Beal in caso di overflow.
Il programma verifica correttamente se un numero è primo oppure no; ed individua correttamente il primo dei fattori primi comuni a tre numeri interi.

Test 1
Operazione scelta: 1
Prima potenza: 5^3
Seconda potenza: 7^9
Terza potenza: 8^10
Equazione: non soddisfatta
Fattore primo comune: nessuno
Esito verifica: verificata

Test 2
Operazione scelta: 1
Prima potenza: 3^3
Seconda potenza: 6^3
Terza potenza: 3^5
Equazione: soddisfatta
Fattore primo comune: 3
Esito verifica: verificata

Test 3
Operazione scelta: 1
Prima potenza: 5^3
Seconda potenza: 5^{10}
Terza potenza: 5^6
Equazione: non soddisfatta
Fattore primo comune: 5
Esito verifica: verificata

Test 4
Operazione scelta: 1
Prima potenza: 729^3
Seconda potenza: 1458^3
Terza potenza: 243^4
Equazione: soddisfatta
Fattore primo comune: 3
Esito verifica: verificata

Test 5
Operazione scelta: 1
Prima potenza: 42^5
Seconda potenza: 24^5
Terza potenza: 60^5
Equazione: non soddisfatta
Fattore primo comune: 2
Esito verifica: verificata

Test 6
Operazione scelta: 1
Prima potenza: 42^3
Seconda potenza: 210^3
Terza potenza: 42^5
Equazione: non soddisfatta
Fattore primo comune: 2
Esito verifica: verificata

Test 7
Operazione scelta: 1
Prima potenza: 7^6
Seconda potenza: 7^7
Terza potenza: 98^3
Equazione: soddisfatta
Fattore primo comune: 7
Esito verifica: verificata

Test 8
Operazione scelta: 1
Prima potenza: 1000^3
Seconda potenza: 1500^3
Terza potenza: 8591^6
Equazione: non soddisfatta
Fattore primo comune: nessuno
Esito verifica: verificata

Test 9
Operazione scelta: 1
Prima potenza: 9^3
Seconda potenza: 27^3
Terza potenza: 81^3
Equazione: non soddisfatta
Fattore primo comune: 3
Esito verifica: verificata

Test 10
Operazione scelta: 1
Prima potenza: 86^3
Seconda potenza: 129^3
Terza potenza: 43^4
Equazione: non soddisfatta
Fattore primo comune: 43
Esito verifica: verificata

Test 11
Operazione scelta: 2
Numero di partenza: 5
Valori ottenuti: 16, 8, 4, 2, 1
Esito verifica: verificata

Test 12
Operazione scelta: 2
Numero di partenza: 13
Valori ottenuti: 40, 20, 10, 5, 16, 8, 4, 2, 1
Esito verifica: verificata

Test 13
Operazione scelta: 2
Numero di partenza: 42
Valori ottenuti: 21, 64, 32, 16, 8, 4, 2, 1
Esito verifica: verificata

Test 14
Operazione scelta: 2
Numero di partenza: 81
Valori ottenuti: 244, 122, 61, 184, 92, 46, 23, 70, 35, 106, 53, 160, 80, 40, 20, 10, 5, 16, 8, 4, 2, 1
Esito verifica: verificata

Test 15
Operazione scelta: 2
Numero di partenza: 47
Valori ottenuti: 142, 71, 214, 107, 322, 161, 484, 242, 121, 364, 182, 91, 274, 137, 412, 206, 103, 310, 155, 466, 233, 700, 350, 175, 526, 263, 790, 395, 1186, 593, 1780, 890, 445, 1336, 668, 334, 167, 502, 251, 754, 377, 1132, 566, 283, 850, 425, 1276, 638, 319, 958, 479, 1438, 719, 2158, 1079, 3238, 1619, 4858, 2429, 7288, 3644, 1822, 911, 2734, 1367, 4102, 2051, 6154, 3077, 9232, 4616, 2308, 1154, 577, 1732, 866, 433, 1300, 650, 325, 976, 488, 244, 122, 61, 184, 92, 46, 23, 70, 35, 106, 53, 160, 80, 40, 20, 10, 5, 16, 8, 4, 2, 1
Esito verifica: verificata

Test 16
Operazione scelta: 2
Numero di partenza: 120
Valori ottenuti: 60, 30, 15, 46, 23, 70, 35, 106, 53, 160, 80, 40, 20, 10, 5, 16, 8, 4, 2
Esito verifica: verificata

Test 17
Operazione scelta: 2
Numero di partenza: 567
Valori ottenuti: 1702, 851, 2554, 1277, 3832, 1916, 958, 479, 1438, 719, 2158, 1079, 3238, 1619, 4858, 2429, 7288, 3644, 1822, 911, 2734, 1367, 4102, 2051, 6154, 3077, 9232, 4616, 2308, 1154, 577, 1732, 866, 433, 1300, 650, 325, 976, 488, 244, 122, 61, 184, 92, 46, 23, 70, 35, 106, 53, 160, 80, 40, 20, 10, 5, 16, 8, 4, 2, 1
Esito verifica: verificata

Test 18
Operazione scelta: 2
Numero di partenza: 2747
Valori ottenuti: 8242, 4121, 12364, 6182, 3091, 9274, 4637, 13912, 6956, 3478, 1739, 5218, 2609, 7828, 3914, 1957, 5872, 2936, 1468, 734, 367, 1102, 551, 1654, 827, 2482, 1241, 3724, 1862, 931, 2794, 1397, 4192, 2096, 1048, 524, 262, 131, 394, 197, 592, 296, 148, 74, 37, 112, 56, 28, 14, 7, 22, 11, 34, 17, 52, 26, 13, 40, 20, 10, 5, 16, 8, 4, 2, 1
Esito verifica: verificata

Test 19
Operazione scelta: 2
Numero di partenza: 1200
Valori ottenuti: 600, 300, 150, 75, 226, 113, 340, 170, 85, 256, 128, 64, 32, 16, 8, 4, 2, 1
Esito verifica: verificata

Test 20
Operazione scelta: 2
Numero di partenza: 30
Valori ottenuti: 15, 46, 23, 70, 35, 106, 53, 160, 80, 40, 20, 10, 5, 16, 8, 4, 2, 1
Esito verifica: verificata

Test 21
Operazione scelta: 3
Primo numero primo: 11
Secondo numero primo: 13
Valore assoluto della differenza: 2
Quadrato del logaritmo naturale del numero minore: 5.75
Esito verifica: verificata

Test 22
Operazione scelta: 3
Primo numero primo: 53
Secondo numero primo: 59
Valore assoluto della differenza: 6
Quadrato del logaritmo naturale del numero minore: 15.76
Esito verifica: verificata

Test 23
Operazione scelta: 3
Primo numero primo: 103
Secondo numero primo: 107
Valore assoluto della differenza: 4
Quadrato del logaritmo naturale del numero minore: 21.48
Esito verifica: verificata

Test 24
Operazione scelta: 3
Primo numero primo: 257
Secondo numero primo: 263
Valore assoluto della differenza: 6
Quadrato del logaritmo naturale del numero minore: 30.79
Esito verifica: verificata

Test 25
Operazione scelta: 3
Primo numero primo: 401
Secondo numero primo: 409
Valore assoluto della differenza: 8
Quadrato del logaritmo naturale del numero minore: 35.93
Esito verifica: verificata

Test 26
Operazione scelta: 3
Primo numero primo: 1279
Secondo numero primo: 1283
Valore assoluto della differenza: 4
Quadrato del logaritmo naturale del numero minore: 51.18
Esito verifica: verificata

Test 27
Operazione scelta: 3
Primo numero primo: 3163
Secondo numero primo: 3167
Valore assoluto della differenza: 4
Quadrato del logaritmo naturale del numero minore: 64.95
Esito verifica: verificata

Test 28
Operazione scelta: 3
Primo numero primo: 1879
Secondo numero primo: 1889
Valore assoluto della differenza: 10
Quadrato del logaritmo naturale del numero minore: 56.83
Esito verifica: verificata

Test 29
Operazione scelta: 3
Primo numero primo: 4931
Secondo numero primo: 4933
Valore assoluto della differenza: 2
Quadrato del logaritmo naturale del numero minore: 72.31
Esito verifica: verificata

Test 30
Operazione scelta: 3
Primo numero primo: 19
Secondo numero primo: 23
Valore assoluto della differenza: 4
Quadrato del logaritmo naturale del numero minore: 8.67
Esito verifica: verificata

6 Verifica del Programma

6.1 Brano di Codice Scelto

```
if (val_corr % 2 == 0)
{
  val_corr = val_corr / 2;
}
else
{
  val_corr = 3 * val_corr + 1;
}
```

6.2 Proprietà da Verificare

Se denotiamo con $val(\texttt{val\_corr})$ il valore contenuto in `val_corr` all'inizio dell'esecuzione, e definita la funzione $f(val(\texttt{val\_corr}))$ come:

$$f(val(\texttt{val\_corr})) = \begin{cases} val(\texttt{val\_corr})/2, & \text{se } val(\texttt{val\_corr}) \text{ pari} \\ 3 \cdot val(\texttt{val\_corr}) + 1, & \text{se } val(\texttt{val\_corr}) \text{ dispari} \end{cases}$$

la proprietà da verificare è che al termine dell'esecuzione risulta:

$$\texttt{val\_corr} = f(val(\texttt{val\_corr}))$$

6.3 Svolgimento

E' possibile verificare la correttezza di istruzioni di selezione tramite regole di Dijkstra. Data la postcondizione

$R_{\texttt{val\_corr}, f(val(\texttt{val\_corr}))} = (\texttt{val\_corr} = f(val(\texttt{val\_corr})))_{\texttt{val\_corr}, f(val(\texttt{val\_corr}))}$

la precondizione più debole affinchè la tripla $\{Q\}S\{R\}$ sia soddisfatta è definita nel modo seguente:

$wp(S,R) = (val(\texttt{val\_corr})\%2 = 0 \ \rightarrow \ wp(S_1,R)) \ \wedge \ (val(\texttt{val\_corr})\%2 \neq 0 \ \rightarrow \ wp(S_2,R))$

$wp(S_1,R) = (val(\texttt{val\_corr})\%2 = 0 \ \rightarrow \ (\texttt{val\_corr} = f(val(\texttt{val\_corr})))_{\texttt{val\_corr}, val(\texttt{val\_corr})/2}) = (val(\texttt{val\_corr})\%2 = 0 \rightarrow val(\texttt{val\_corr})/2 = f(val(\texttt{val\_corr}))) \equiv vero$

$wp(S_2,R) = (val(\texttt{val\_corr})\%2 \neq 0 \ \rightarrow \ (\texttt{val\_corr} = f(val(\texttt{val\_corr})))_{\texttt{val\_corr}, 3*val(\texttt{val\_corr})+1}) = (val(\texttt{val\_corr})\%2 \neq 0 \rightarrow 3 * val(\texttt{val\_corr}) + 1 = f(val(\texttt{val\_corr}))) \equiv vero$

$wp(S,R) = (vero \wedge vero) \equiv vero$

Il brano di codice scelto produce dunque il risultato corretto in ogni caso, a prescindere dallo stato iniziale della computazione.